Novel Adaptive Equalization Techniques for a Transmit Diversity Scheme

Yan Zeng

A thesis submitted in partial fulfilment
of the requirements for the degree of
Master of Engineering
in
Electrical and Electronic Engineering
at the
University of Canterbury,
Christchurch, New Zealand.

November 2006

ABSTRACT

Space-time block coding (STBC) has added a new dimension to broadband wireless communication systems. Applications such as wireless Internet access and multimedia require the transmission of high data rates over frequency selective fading channels. The reliability of the wireless communication system can be increased by using diversity techniques combined with an equalizer at the receiver to eliminate the inter-symbol interference caused by multipath propagation. Generalizing Alamouti's famous STBC method to frequency selective channels, Time Reversal-Space Time Block Coding (TR-STBC) was first introduced in [1] and has since been shown to be an effective transmit diversity scheme [2, 3, 4]. TR-STBC-based schemes are considered promising candidates for indoor transmission [5] as well as for the enhanced data rates of the global evolution (EDGE) system [2, 3].

The optimal equalizer for a TR-STBC-based transceiver is the Maximum Likelihood Sequence Estimator (MSLE), realized using the Viterbi algorithm. Unfortunately, a Viterbi equalizer is difficult to implement in real-time due its exponential increase in complexity with the number of antennas and the length of the channel impulse response. Thus, we consider an adaptive algorithm-based Decision Feedback Equalizer (DFE). Such a DFE requires only linear processing complexity while maintaining good performance.

Theoretically, the two output streams of a 2×1 TR-STBC decoder are uncoupled in terms of the input signal statistics and uncorrelated in terms of the channel noise statistics. The standard approach to removing the inter-symbol interference from these streams is to use either two parallel independently-adapted Single-Input Single-Output (SISO) equalizers or to use a single Multiple-Input Multiple-Output (MIMO) equalizer. By exploiting the common second-order statistics of the two output streams, we propose

a novel hybrid equalizer structure which shares the statistical information between two SISO equalizers while constraining them to have common tap weights.

To accommodate various levels of performance versus computational complexity, we propose novel Least Mean Square (LMS), Normalized Least Mean Square (NLMS), and Recursive Least Squares (RLS)-based adaptive algorithms for this new equalizer architecture. We use both statistical analysis and Monte Carlo simulations to characterize the dynamic convergence of these algorithms and to compare our new structure with the conventional uncoupled SISO equalizers and fully-coupled MIMO equalizer. We show that our new equalizer outperforms the other two equalizers using a reduced computational complexity similar to the uncoupled SISO equalizers. As expected, with increasing complexity, we find that the novel RLS-based algorithms converge the fastest followed by the novel NLMS- and LMS-based algorithms.

We also consider alternative packet structures and kick-start methods to increase the convergence speed and reliability of the equalizer at realistic complexity. Finally, adding multiple receiver antennas to our system, we extend our equalizer structures and algorithms to the $2 \times N_R$ case. Using analysis and simulations, we demonstrate that the added receiver diversity in this case yields even greater reliability.

ACKNOWLEDGEMENTS

There are many people I must acknowledge for assisting me on this long and rocky road to completing my thesis. My biggest thanks goes out to my supervisor, Dr. Lee Garth, whose constant guidance and support (even from oceans away) saw me through the at times seemingly impossible task of finishing.

Much gratitude goes out to my parents, your love and support made my Masters journey that much easier. Thanks to my cousin for putting up with my late nights and lack of cooking.

I would like to thank my former colleagues/friends at Tait Electronics, for your help and support during my part-time thesis, and providing me with much needed reference materials.

Thanks to my friends, especially Beth, Carrie, Malinda and Tong, for believing in me, keeping me in line and sharing the good and bad times.

Finally, Rio, thank you for looking over my shoulder.

CONTENTS

Chapter 1

INTRODUCTION

1.1 SPACE-TIME DIVERSITY TECHNIQUES FOR WIRELESS CHANNELS

The demand for capacity in cellular and wireless local area networks has grown exponentially over the last decade. Wireless Internet access and multimedia applications require a huge increase in information throughput whilst maintaining reliability. However, in most situations the wireless channel suffers spatially selective fading due to multipath propagation. One way to prevent the resulting performance degradation is to make several replicas of the signal available to the receiver to increase the probability that at least some of them are not severely attenuated. This technique is called diversity combining, which can be provided using temporal, frequency, polarization and spatial redundancy.

As the number of replicas or diversity paths increases, it has been shown that the system error performance over a fading channel goes from an algebraically decreasing function of the signal-to-noise ratio (SNR) to an exponential one [6]. This can be written

$$P_e \sim \underbrace{\frac{1}{1+\frac{E_{\text{avg}}}{N_o}}}_{\text{No diversity}} \xrightarrow{N\to\infty} \underbrace{\frac{1}{\left(1+\frac{E_{\text{avg}}}{NN_o}\right)^N}}_{\text{Large diversity}} \to e^{-\frac{E_{\text{avg}}}{N_o}} \tag{1.1}$$

where $\frac{E_{\text{avg}}}{N_o}$ is the average SNR and N is the diversity order.

Recently, new space diversity techniques have been developed, using multiple an-

tennas at both the transmitter and the receiver. In particular, transmit diversity techniques have been invented which harness the power of MIMO systems (Fig. 1.1) to increase the reliability of wireless transceivers. Fortunately, implementing complex MIMO signal processing algorithms has become feasible due to the exponentially increasing computational power of modern integrated circuits [7].

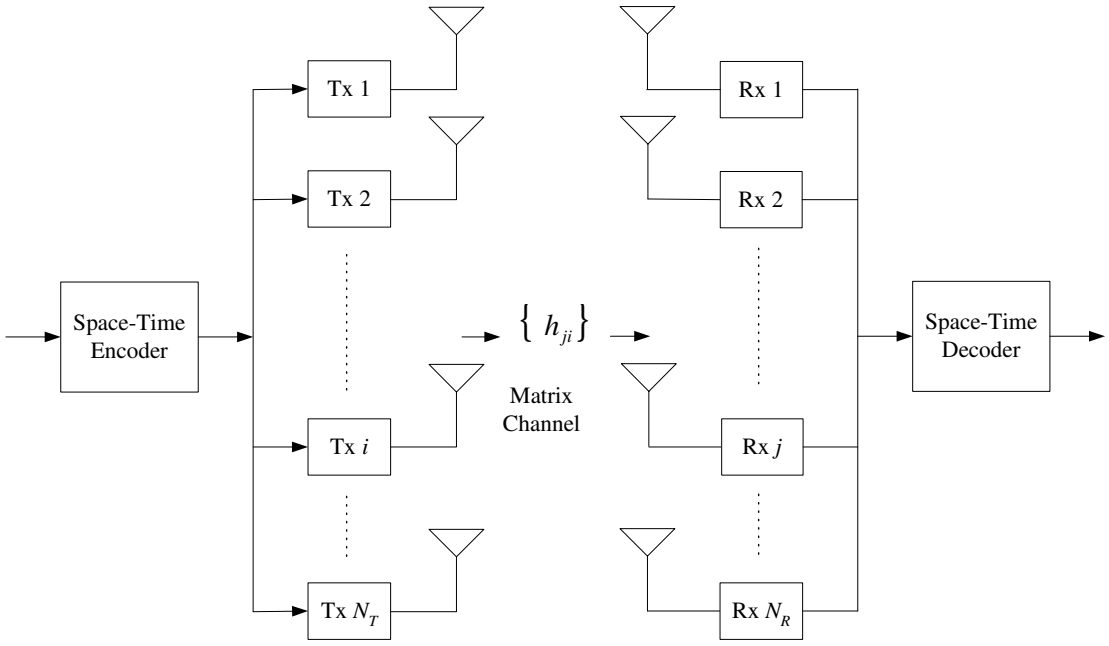

Figure 1.1 General Space-Time Architecture

The concept of space-time diversity is as follows. The wireless channel usually varies with time, particularly for mobile systems. This fading in the received signal amplitude is often modeled using a Rayleigh distribution. So, with a given probability the channel gain can sometimes be so small that the channel becomes useless. Space-time diversity helps mitigate this problem by having the transmitter send the same information over multiple channels which fade independently of each other. In the space dimension, this is accomplished by using multiple antennas at the transmitter and / or receiver. In the time dimension, this is achieved by repeating the transmission multiple times.

Transmit diversity (TD) techniques, using multiple transmit antennas in the base station of a cellular system or wireless local area network, have attracted special interest. This is primarily due to the performance gains they yield without requiring extra

receive antennas in the mobile, which would increase the power consumption due to the added complexity. Also, the cost of multiple transmit antennas at the base station can be shared among many users.

Other advantages of TD techniques are as follows. They can be easily combined with channel coding to realize a coding gain as well as a diversity gain [8]. TD techniques also do not require channel state information (CSI) at the transmitter, eliminating the additional hardware required to support a reverse link. Finally, TD techniques have been shown to be robust against real world conditions such as antenna correlation, channel estimation errors, and Doppler effects [9].

1.2 PRACTICAL EQUALIZERS FOR TR-STBC SYSTEMS

The goal of this research is to devise and study practical equalizer structures for TD-oriented MIMO systems. In particular, we concentrate on constructing equalizers for the TR-STBC scheme, which was introduced in [1] as an extension of the Alamouti scheme [10] to frequency selective channels. As we will show in the next chapter, if N_T transmit antennas are used, the TR-STBC decoder produces N_T output streams that are uncoupled in terms of the input signal statistics and uncorrelated in terms of the channel noise statistics. For a frequency selective channel, each of these streams contains inter-symbol interference (ISI), which must be mitigated using equalization. The standard approach is to use either N_T parallel independently-adapted Single-Input Single-Output (SISO) equalizers or to use a single Multiple-Input Multiple-Output (MIMO) equalizer.

The optimal sequence detection method for the streams is the Maximum Likelihood Sequence Estimator (MLSE), implemented using the Viterbi algorithm. Unfortunately, the computational complexity of the Viterbi equalizer increases exponentially with the channel delay spread and the transmitted signal constellation size. Thus, with our emphasis on real-time implementation, we consider here alternative lower complexity adaptive equalization methods. An adaptive algorithm-based DFE is the fundamental architecture for our equalizer designs. Such a DFE requires only linear processing complexity while maintaining good performance.

By exploiting the common second-order statistics of the N_T output streams, we propose a new hybrid equalizer structure which shares the statistical information between N_T SISO equalizers while constraining them to have common tap weights. To accommodate various levels of performance versus computational complexity, we propose novel Least Mean Square (LMS), Normalized Least Mean Square (NLMS), and Recursive Least Squares (RLS)-based adaptive algorithms for this new equalizer architecture. We use both statistical analysis and Monte Carlo simulations for the 2×1 case to characterize the dynamic convergence of these algorithms and to compare our new structure with the conventional uncoupled SISO equalizers and fully-coupled MIMO equalizer. We show that our new equalizer outperforms the other two equalizers using a reduced computational complexity similar to the uncoupled SISO equalizers. As expected, with increasing complexity, we find that the novel RLS-based algorithms converge the fastest followed by the novel NLMS- and LMS-based algorithms.

We also consider alternative packet structures and kick-start methods to increase the convergence speed and reliability of the equalizer at realistic complexity. Finally, adding multiple receiver antennas to our system, we extend our equalizer structures and algorithms to the $2 \times N_R$ case. Using analysis and simulations for a variety of wireless channels, we demonstrate that the added receiver diversity in this case yields even greater reliability.

In our study we consider orthogonal Space Time Block Codes (STBCs) because of their linear decoding complexity. Our motivation for mainly concentrating on the two transmit antenna case is as follows. For $N_T = 2$, the data rate is preserved at $R = 1$, which enables retrofitting of existing systems [5]. Unfortunately, as N_T increases, the data rate R drops quickly. Also, the SNR gain suffers "diminishing returns" for the increased complexity [8]. Although non-orthogonal STBCs can be used to improve the data rate for $N_T > 2$, the complexity of the corresponding decoder becomes polynomial or exponential. Therefore, we feel that two transmit antennas provide the best trade-off between rate and decoding complexity for a given diversity gain. On the receiver side, receiver diversity techniques are well known and well studied. We first consider the more practical case of one receive antenna on the mobile followed by the general case

of N_R receive antennas.

Throughout our research a key focus is on the practical complexity issues leading to real-time system implementation in a MIMO testbed similar to that of Tait Electronics's Space-Time Array Research (STAR) group [11]. Therefore, as we consider each new architecture and algorithm, we determine its real-time implementation complexity. Our goal is to find methods with the right balance between good performance and realistic complexity, enabling real-time processing and practical implementation in embedded hardware.

1.3 LITERATURE REVIEW OF TR-STBC SYSTEMS AND EQUALIZATION SCHEMES

The earliest form of spatial transmit diversity is the delay diversity scheme proposed by Wittneben [12], where the same information is transmitted from two antennas simultaneously but with a delay of one symbol interval. The next major breakthrough in TD came in the form of the Alamouti scheme [10]. The very simple two transmit antenna structure and linear processing of the Alamouti scheme has been adopted into both the third generation cellular standards W-CDMA and CDMA-2000 [13]. This scheme has been generalized to an arbitrary number of transmit antennas by many researchers (see e.g., [14]).

Researchers have also extended the Alamouti STBC-based scheme for flat fading channels to frequency-selective channels. Most of the schemes are designed for quasi-static block-fading channels (where the channel is static during a data block but varies from one block to the next), which inherently limit their applications to lower mobility scenarios. Three well-known orthogonal STBC structures for the frequency selective channel have been proposed: Single Carrier TR-STBC [1], Orthogonal Frequency Division Multiplexing-STBC (OFDM-STBC) [15], and Single Carrier Frequency Domain Equalization-STBC (SC FDE-STBC) [16, 17]. TR-STBC includes a time domain equalization scheme while the latter two rely on frequency domain-based equalizers utilizing the Fast Fourier Transform (FFT). TR-STBC is the most suitable diversity technique for time division multiple access (TDMA)-based communication systems such as the

global system for mobile communications (GSM) and the enhanced data rates for GSM evolution (EDGE).

Several researchers have considered the performance of TR-STBC relative to the other transmit diversity schemes. A comparison study done for the EDGE Typical Urban (TU) channel by Al-Dhahir on TR-STBC, OFDM-STBC, and FDE-STBC showed that TR-STBC with a DFE can outperform both the FDE-STBC and OFDM-STBC schemes [2, 3]. Al-Dhahir *et al.* have also shown that TR-STBC is less sensitive to channel estimation errors and outperforms other high-complexity STBC methods [4]. They have demonstrated that for TR-STBC a suboptimal MMSE-DFE can achieve near-optimum performance with as few as three feedback taps.

A number of researchers have proposed STBC schemes with optimal MLSE equalization [18]. Most, however, have acknowledged that low complexity suboptimum equalizers for STBC's are essential. Modifying the optimal MLSE, Schober *et al.* [19] have proposed a Decision-Feedback Sequence Estimation (DFSE) equalizer for TR-STBC transmission in GSM and EDGE systems. Their DFSE is a detection algorithm based on a trellis with an adjustable number of states, which when set to the maximum value is equivalent to the MLSE-based Viterbi algorithm, and when set to zero reduces to the decision-feedback detector. Their simulation results show that a whitened version of the DFSE with a small number of states yields the best performance. Recently, Gerstaker *et al.* [5] have investigated TR-STBC followed by a DFE in the context of the IEEE 802.11b WLAN standard for indoor wireless channels. Their Monte Carlo simulations for a 2×2 configuration show an improvement of 7dB over a conventional SISO configuration for an indoor frequency selective channel. The STAR research team at Tait Electronics Ltd. has also studied the complexity of channel estimation and real-time hardware implementation using a TR-STBC-based MIMO testbed [11, 20, 21] with a Viterbi equalizer.

Regarding suboptimal equalizers for 2×2 TR-STBC systems, other researchers have considered a RLS equalizer for 2×2 TR-STBC, leading to good simulated results but extremely high computational complexity [22]. Finally, a blind equalization scheme for 2×2 TR-STBC using the Constant Modulus Algorithm (CMA) has been proposed [23].

1.4 THESIS OUTLINE

This thesis is organized as follows. In Chapter 2 we review the Alamouti scheme [10] and its extension to the TR-STBC scheme for frequency selective channels [1]. We also summarize the adaptive equalization methods used in this research: the Wiener filter and Method of Steepest Descent (SD), the LMS, NLMS, and RLS algorithms, and the standard decision feedback equalizer (DFE) architecture. We finally introduce the statistical analysis methods applied in this thesis: the Independence Method and the Small Step-size Method. In Chapter 3, first we tailor the standard SISO DFE to mitigate the residual ISI in each decoupled TR-STBC decoder output stream. We then propose a novel equalizer architecture, exploiting shared channel knowledge between the TR-STBC decoder outputs to constrain the tap update algorithm, leading to faster convergence and improving the equalizer error performance. We compare the conventional SISO DFE against the novel hybrid DFE and the MIMO DFE adapted using customized versions of the LMS, NLMS and RLS algorithms. We then introduce a novel prototype TR-STBC packet structure and highlight its advantages over the conventional TR-STBC packet structure. We also discuss practical parameter values such as packet length to accommodate the characteristics of the target wireless channel. To speed up the DFE convergence, using an estimate of the CSI, we apply the Method of SD to jump-start the DFE in the TR-STBC detector. In Chapter 4, adding multiple receive antennas, we extend the 2×1 architecture with SISO DFEs to a $2 \times N_R$ architecture with MISO DFEs. Using Monte Carlo simulations, we compare the various equalizer structures and algorithms for 2×2 and 2×4 systems operating over a variety of wireless channels. In Chapter 5, we conclude with the core results of this thesis and indicate possible topics for future research.

Chapter 2

BACKGROUND

2.1 INTRODUCTION TO ORTHOGONAL SPACE TIME BLOCK CODES

Orthogonal space time block codes (OSTBCs) are an important subclass of linear STBCs. They have the property that the received data is decoupled and the optimal maximum likelihood (ML) decoding process is linear. OSTBCs also achieve the maximum possible transmission rate for the full diversity order of $N_R \times N_T$ [7].

Note that STBCs are not really codes in the conventional sense, but rather are multi-dimensional constellation designs. A STBC is defined by a $p \times N_T$ transmission matrix $\mathcal{G}$. The p rows of the matrix represent different time slots, whereas the N_T columns of the matrix correspond to different transmit antennas. The entries of $\mathcal{G}$ are linear combinations of the transmitted symbols $x_1, x_2, \ldots, x_k$ and their conjugates. The rate of the STBC is given by $R = k/p$, since p time slots are used to transmit the k symbols. For example, a rate one code which utilizes two transmit antennas can be defined by

$$\mathcal{G}_2 = \begin{pmatrix} x_1 & x_2 \\ -x_2^* & x_1^* \end{pmatrix}. \tag{2.1}$$

Given that the transceiver employs a 2^b-point signal constellation, during a single time slot, kb bits arrive at the encoder, corresponding to constellation symbols $s_1, \ldots, s_k$. Setting $x_i = s_i$ for $i = 1, \ldots, k$ in $\mathcal{G}$, we obtain a code matrix $\boldsymbol{C}$ containing specific constellations symbols which are transmitted from the N_T antennas. Matrix element $[\boldsymbol{C}]_{ti}$ contains the symbol transmitted at time slot t from antenna i. The t-th row of $\boldsymbol{C}$ contains the symbols transmitted simultaneously by the N_T transmit

antennas during time slot t.

The original general derivation of OSTBCs is due to Tarokh *et al.* [14], who studied the error performance associated with $\boldsymbol{C}$ matrices that are unitary. They found that the orthogonality of the columns of $\boldsymbol{C}$ enables the data streams from the different transmitters to be decoupled and allows the maximum likelihood decoding process of the OSTBC to be linear.

The orthogonality design criterion has been motivated and derived in a number of different ways since [14]. For example, changing the notation slightly, let us define $\boldsymbol{X} = \boldsymbol{C}^T$, where $(\cdot)^T$ represents the matrix transpose operation. For a set of k transmitted symbols, an OSTBC is a linear STBC whose code matrix transpose $\boldsymbol{X}$ has the special unitary property [7]

$$\boldsymbol{X}\boldsymbol{X}^H = \sum_{n=1}^{k} |s_n|^2 \cdot \boldsymbol{I} \tag{2.2}$$

where $(\cdot)^H$ denotes the Hermitian transpose. Note that the identity matrix on the righthand side of (2.2) can be scaled by any arbitrary constant factor.

To see why setting $\boldsymbol{X}$ to be proportional to a unitary matrix guarantees decoupled detection, consider the following general STBC code matrix:

$$\boldsymbol{X} = \sum_{n=1}^{k} \mathrm{Re}\{s_n\}\, \boldsymbol{A}_n + i\ \mathrm{Im}\{s_n\}\, \boldsymbol{B}_n \tag{2.3}$$

where $\{\boldsymbol{A}_n,\ \boldsymbol{B}_n\}$ are fixed $N_T \times p$ unitary matrices with real elements, and $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ are the real and imaginary parts, respectively. Let $\boldsymbol{Y}$ be the $N_R \times p$ received data matrix and $\boldsymbol{H}$ be the $N_R \times N_T$ channel matrix. ML detection for STBCs amounts to minimizing the metric [7]

$$\begin{aligned}
\|\boldsymbol{Y} - \boldsymbol{H}\boldsymbol{X}\|^2 &= \|\boldsymbol{Y}\|^2 - 2\,\mathrm{Re}\{\mathrm{Tr}(\boldsymbol{Y}^H\boldsymbol{H}\boldsymbol{X})\} + \|\boldsymbol{H}\boldsymbol{X}\|^2 \\
&= \|\boldsymbol{Y}\|^2 - 2\sum_{n=1}^{k} \mathrm{Re}\{\mathrm{Tr}(\boldsymbol{Y}^H\boldsymbol{H}\boldsymbol{A}_n)\}\,\mathrm{Re}\{s_n\} \\
&\quad + 2\sum_{n=1}^{k} \mathrm{Im}\{\mathrm{Tr}(\boldsymbol{Y}^H\boldsymbol{H}\boldsymbol{B}_n)\}\,\mathrm{Im}\{s_n\} + \|\boldsymbol{H}\|^2 \cdot \|\boldsymbol{s}\|^2
\end{aligned}$$

where $||\cdot||$ and $\mathrm{Tr}(\cdot)$ denote the Frobenius norm and trace of a matrix. Bringing the last three terms into a single sum, we have

$$\begin{aligned}\|\boldsymbol{Y}\|^2 + \sum_{n=1}^{k} \big[& -2\,\mathrm{Re}\{\mathrm{Tr}(\boldsymbol{Y}^H \boldsymbol{H} \boldsymbol{A}_n)\}\,\mathrm{Re}\{s_n\} \\ & + 2\,\mathrm{Im}\{\mathrm{Tr}(\boldsymbol{Y}^H \boldsymbol{H} \boldsymbol{B}_n)\}\,\mathrm{Im}\{s_n\} + \|\boldsymbol{H}\|^2 \cdot |s_n|^2 \big] \\ & = \|\boldsymbol{H}\|^2 \cdot \sum_{n=1}^{k} \left| s_n - \frac{\mathrm{Re}\{\mathrm{Tr}(\boldsymbol{Y}^H \boldsymbol{H} \boldsymbol{A}_n)\} - i\,\mathrm{Im}\{\mathrm{Tr}(\boldsymbol{Y}^H \boldsymbol{H} \boldsymbol{B}_n)\}}{\|\boldsymbol{H}\|^2} \right|^2 . \end{aligned} \tag{2.4}$$

The equality follows from the unitary property of $\boldsymbol{A}_n$ and $\boldsymbol{B}_n$, which yields

$$[\mathrm{Re}\{\mathrm{Tr}(\boldsymbol{Y}^H \boldsymbol{H} \boldsymbol{A}_n)\}]^2 + [\mathrm{Im}\{\mathrm{Tr}(\boldsymbol{Y}^H \boldsymbol{H} \boldsymbol{B}_n)\}]^2 = \|\boldsymbol{Y}\|^2 \cdot \|\boldsymbol{H}\|^2 .$$

Equation (2.4) demonstrates that the ML metric reduces to a sum of k terms, where each term depends on only one complex symbol. Thus, the detection of s_n is decoupled from the detection of s_p for $n \neq p$.

In addition, Tarokh *et al.* proved that complex OSTBCs of rate $R = 1$ exist only for $N_T = 2$. Unfortunately, for $N_T > 2$, the rate of OSTBCs decreases below $R = 1$, reducing the bandwidth efficiency of the system [14]. As an alternative, quasi-orthogonal STBCs can preserve the full diversity and full rate at the cost of a small loss in BER performance and some extra decoding complexity relative to truly orthogonal schemes [24].

2.2 INTRODUCTION TO TIME-REVERSAL SPACE TIME BLOCK CODES

Alamouti's Space Time Block Coding (STBC) scheme is a transmit diversity scheme, where, by utilizing two transmit and one receive antennas, the transmitter produces the same diversity benefit at the receiver as can be achieved by using one transmit and two receive antennas in a conventional Maximal Ratio Receive Combining (MRRC) scheme [10]. Thus, the transmitter diversity is transformed into "virtual" receive diversity, given knowledge of the channel state information (CSI) and linear decoding at the receiver.

Table 2.1 Alamouti transmission schedule

	TX 1	TX 2
Time k	$s_1(k)$	$s_2(k)$
Time $k+1$	$-s_2^*(k)$	$s_1^*(k)$

Let us review the scheme proposed by Alamouti for a flat fading channel, summarized in Table 2.1. We assume that the channel is static for two consecutive symbol times (known as quasi-static). The original symbol stream is divided into two separate streams, $s_1(k)$ and $s_2(k)$. These two symbol streams are then transmitted on antenna 1 and antenna 2 in alternating time slots. In even time slots, $s_1(k)$ is transmitted from antenna 1 and $s_2(k)$ is transmitted from antenna 2. Thus, for each even time slot, the signal received by the single antenna has the form

$$r_1(k) = h_1\, s_1(k) + h_2\, s_2(k) + n_1(k) \tag{2.5}$$

where h_1 is the channel between transmit antenna 1 and the receive antenna, and h_2 is the channel between transmit antenna 2 and the receiver. Variable $n_1(k)$ represents the "even" noise sample. If we let $(\cdot)^*$ denote complex conjugation, in odd time slots, $s_1^*(k)$ is transmitted from antenna 2 and $-s_2^*(k)$ is transmitted from antenna 1. Thus, the received observation for the odd time slot is

$$r_2'(k) = h_2\, s_1^*(k) - h_1\, s_2^*(k) + n_2^*(k) \tag{2.6}$$

where $n_2^*(k)$ represents the odd noise sample. Complex conjugating the odd received observation, we have

$$r_2(k) = [r_2'(k)]^* = h_2^*\, s_1(k) - h_1^*\, s_2(k) + n_2(k)\,. \tag{2.7}$$

We can rewrite (2.5)–(2.7) into blocks using the following matrix and vector notation,

$$\boldsymbol{r}(k) = \begin{bmatrix} r_1(k) \\ r_2(k) \end{bmatrix}, \qquad \boldsymbol{s}(k) = \begin{bmatrix} s_1(k) \\ s_2(k) \end{bmatrix}, \qquad \boldsymbol{n}(k) = \begin{bmatrix} n_1(k) \\ n_2(k) \end{bmatrix}, \tag{2.8}$$

and the single-tap channel matrix

$$\boldsymbol{H} = \begin{bmatrix} h_1 & h_2 \\ h_2^* & -h_1^* \end{bmatrix}. \tag{2.9}$$

Thus, the received observation blocks have the form

$$\boldsymbol{r}(k) = \boldsymbol{H}\boldsymbol{s}(k) + \boldsymbol{n}(k)\,. \tag{2.10}$$

In [10], as a linear decoder, Alamouti multiplies $\boldsymbol{r}(k)$ by $\boldsymbol{H}^H$, producing a decoupled matched filtered output block

$$\boldsymbol{z}(k) = \boldsymbol{H}^H\boldsymbol{r}(k) = \boldsymbol{H}^H\boldsymbol{H}\boldsymbol{s}(k) + \boldsymbol{H}^H\boldsymbol{n}(k) = \left(|h_1|^2 + |h_2|^2\right)\boldsymbol{s}(k) + \boldsymbol{v}(k) \tag{2.11}$$

where

$$\boldsymbol{v}(k) = \boldsymbol{H}^H\,\boldsymbol{n}(k)\,. \tag{2.12}$$

The components of $\boldsymbol{z}(k)$ have the form

$$\begin{aligned} z_1(k) &= \left(|h_1|^2 + |h_2|^2\right)s_1(k) + v_1(k) \\ z_2(k) &= \left(|h_1|^2 + |h_2|^2\right)s_2(k) + v_2(k)\,. \end{aligned} \tag{2.13}$$

Note that both components of $\boldsymbol{z}(k)$ experience the diversity gain equivalent to that of MRRC. Regarding the noise components $v_1(k)$ and $v_2(k)$, because they are jointly Gaussian and uncorrelated, they are independent. Thus, for the ideal case of perfect channel knowledge at the receiver, the decoupled matched filter outputs $z_1(k)$ and $z_2(k)$ can then be independently processed to estimate $s_1(k)$ and $s_2(k)$. On the other hand, for the practical case of imperfect channel estimates at the receiver, the matched filter outputs in general will be coupled. In this case, joint processing of $z_1(k)$ and $z_2(k)$ could be beneficial and more robust to the loss of orthogonality.

As shown by Linskog and Paulraj [1], the same signal processing approach can be applied to frequency selective channels, where the channels are modeled by causal

Table 2.2 TR-STBC transmission schedule

	TX 1	TX 2
Time t	$s_1(t)$	$s_2(t)$
Time $t+T$	$-\widetilde{s}_2^*(t)$	$\widetilde{s}_1^*(t)$

discrete-time filters $h_1(q^{-1})$ and $h_2(q^{-1})$. Here we represent the filters using a polynomial in the unit delay operator q^{-1} with the following function

$$\begin{aligned} a(q^{-1})\,u(t) &= (a_0 + a_1\,q^{-1} + \cdots + a_{N_a-1}\,q^{-N_a+1})\,u(t) \\ &= a_0\,u(t) + a_1\,u(t-1) + \cdots + a_{N_a-1}\,u(t-N_a+1)\,. \end{aligned}$$

The complex conjugate of the filter $a(q^{-1})$ is given by

$$[a(q^{-1})]^* \stackrel{\triangle}{=} a^*(q) = a_0^* + a_1^*\,q + \cdots + a_{N_a-1}^*\,q^{N_a-1}$$

which is *anti-causal*. Thus, the anti-causal (time-reversed) counterparts of the two channel filters are $h_1^*(q)$ and $h_2^*(q)$. Note that the flat fading Alamouti scheme corresponds to the single-tap channels $h_1(q^{-1}) = h_1$ and $h_2(q^{-1}) = h_2$.

For the frequency selective channel, instead of transmitting on a symbol basis, this time we transmit on a blockwise basis. Given a block size M, the analogous quantities in discrete time are

$$\begin{aligned} s_1(t) &= [d_1(0),\, d_1(1),\, \ldots,\, d_1(M-1)] \\ s_2(t) &= [d_2(0),\, d_2(1),\, \ldots,\, d_2(M-1)] \\ -\widetilde{s}_2^*(t) &= [-d_2^*(M-1),\, -d_2^*(M-2),\, \ldots,\, -d_2^*(0)] \\ \widetilde{s}_1^*(t) &= [d_1^*(M-1),\, d_1^*(M-2),\, \ldots,\, d_1^*(0)] \end{aligned} \tag{2.14}$$

where $d_i(j)$ is the j-th symbol of the i-th data stream and $\widetilde{s}(t)$ denotes the time reversal of $s(t)$. Table 2.2 shows the TR-STBC transmission schedule. Given this transmission

schedule, during the first (even) transmission interval t, the receiver observes

$$r_1(t) = h_1(q^{-1})\, s_1(t) + h_2(q^{-1})\, s_2(t) + n_1(t)\,. \tag{2.15}$$

Whereas during the second (odd) time interval $t+T$, the receiver measures

$$r_2'(t) = h_2(q^{-1})\, \widetilde{s}_1^*(t) - h_1(q^{-1})\, \widetilde{s}_2^*(t) + n_2(t)\,. \tag{2.16}$$

The second burst of observation samples $r_2'(t)$ is then time-reversed and complex conjugated at the receiver

$$r_2(t) = [\widetilde{r}_2'(t)]^* = h_2^*(q)\, s_1(t) - h_1^*(q)\, s_2(t) + \widetilde{n}_2^*(t)\,. \tag{2.17}$$

Again, we can rewrite (2.15)–(2.17) using the following matrix and vector notation

$$\boldsymbol{r}(t) = \begin{bmatrix} r_1(t) \\ r_2(t) \end{bmatrix}, \qquad \boldsymbol{H}(t) = \begin{bmatrix} h_1(q^{-1}) & h_2(q^{-1}) \\ h_2^*(q) & -h_1^*(q) \end{bmatrix}. \tag{2.18}$$

Multiplying $\boldsymbol{r}(t)$ by $\boldsymbol{H}^H$ again produces decoupled matched filter outputs. To demonstrate this, because we have

$$\boldsymbol{H}^H(t) = \begin{bmatrix} h_1^*(q) & h_2(q^{-1}) \\ h_2^*(q) & -h_1(q^{-1}) \end{bmatrix}, \tag{2.19}$$

the output of the TR-STBC decoder becomes

$$\boldsymbol{z}(t) = \boldsymbol{H}^H(t)\, \boldsymbol{r}(t) = \boldsymbol{H}^H(t)\, \boldsymbol{H}(t)\, \boldsymbol{s}(t) + \boldsymbol{H}^H(t)\, \boldsymbol{n}(t) \tag{2.20}$$

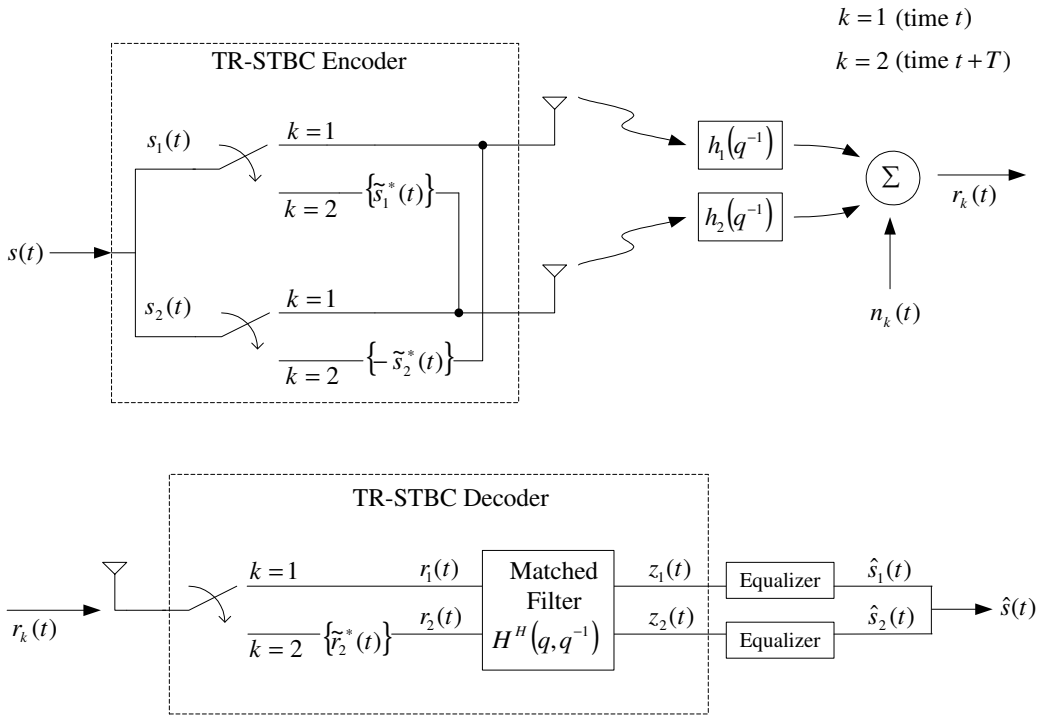

Figure 2.1 Block diagram of 2×1 TR-STBC system

where

$$
\begin{aligned}
z_1(t) &= h_1^*(q)\, r_1(t) + h_2(q^{-1})\, r_2(t) \\
&= \left[h_1^*(q)\, h_1(q^{-1}) + h_2(q^{-1})\, h_2^*(q)\right] s_1(t) + h_1^*(q)\, n_1(t) + h_2(q^{-1})\, \widetilde{n}_2^*(t) \qquad (2.21) \\
z_2(t) &= h_2^*(q)\, r_1(t) - h_1(q^{-1})\, r_2(t) \\
&= \left[h_1(q^{-1})\, h_1^*(q) + h_2^*(q)\, h_2(q^{-1})\right] s_2(t) + h_2^*(q)\, n_1(t) - h_1(q^{-1})\, \widetilde{n}_2^*(t)\,. \qquad (2.22)
\end{aligned}
$$

Figure 2.1 gives a broad outline of the TR-STBC transmit, receive and equalization process.

2.3 A BRIEF OVERVIEW OF ADAPTIVE EQUALIZATION TECHNIQUES

The optimal equalizer for a frequency selective additive white Gaussian noise (AWGN) channel is the maximum likelihood sequence estimator (MLSE), which employs the Viterbi algorithm [25]. The Viterbi algorithm is a trellis structure-based decoder, which

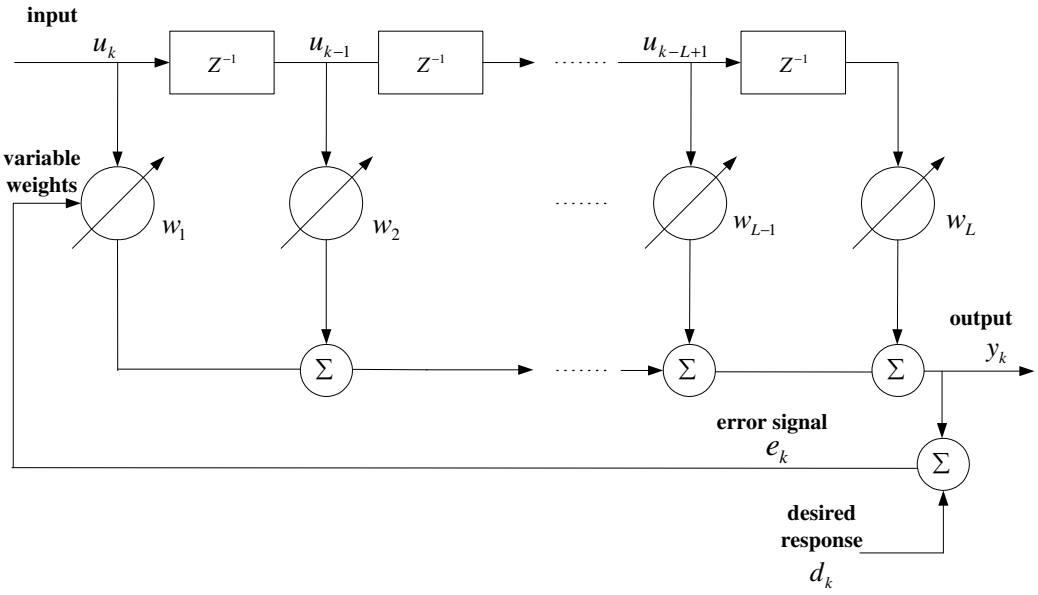

Figure 2.2 Block diagram of adaptive linear equalizer

utilizes the *a posteriori* probability density functions of the signals and computes the traceback paths and the state metrics corresponding to different possible transmitted symbol sequences. The metrics are based on the Euclidean distance. An MLSE-based equalizer yields the best possible theoretical performance but is computationally intensive. A linear increase in the constellation size and channel delay spread results in an exponential increase in the number of trellis states.

Fortunately, there are suboptimal equalizers which can deliver near optimal performances with only a linear increase in computational complexity. For zero-mean i.i.d data, they are optimal in terms of the first and second moments instead of the probability distributions of the received signals. The suboptimal equalizers often minimize a cost function based on the average squared difference between the equalizer outputs and the transmitted symbols. This cost function is known as the Mean Squared Error (MSE) [25].

Generally, equalizer structures fall into two categories: linear and nonlinear. A linear equalizer may be implemented using the Finite Impulse Response (FIR) filter shown in Fig. 2.2 with adjustable coefficients tuned using an adaptive algorithm. For example, the well-known LMS algorithm uses a stochastic gradient method to minimize

the MSE. Note that the optimum minimum mean-square error (MMSE) solution for the equalizer coefficients is given by the Wiener FIR filter solution. Thus, we introduce Wiener filter theory before discussing the LMS algorithm in the next sections. We also review the classical NLMS and RLS algorithms. Two widely-used nonlinear equalizers are the DFE, which we will discuss, and the MLSE using the Viterbi algorithm.

There are typically two modes of operation for an adaptive equalizer: training mode and decision-directed mode. During training mode, a known Pseudo-Noise (PN) sequence is transmitted, and a synchronized version of it is generated in the receiver, which is fed to the equalizer as the desired response d_k. Using the difference between d_k and the actual equalizer output y_k, the equalizer tap weights $w_1, \ldots, w_L$ are adapted to remove the distortion introduced by the channel. After this training sequence, the equalizer switches to decision-directed mode when new data is transmitted to the receiver. To track changes in the channel, the equalizer tap weights are adapted using the output of a decision device as the reference instead of the training sequence. Because the decisions made by the decision device are correct most of the time, the adaptive equalizer is able to track the channel while mitigating its distortion.

2.3.1 Wiener Filter

Before introducing truly adaptive algorithms, we review the Wiener filter, which provides a theoretical benchmark, minimizing the MSE given complete knowledge of the channel statistics. In 1942 Wiener studied the continuous-time linear filtering problem and derived the famous Wiener-Hopf integral equation [26]. The discrete-time equivalent of the integral equation is known as the normal equation. Both the continuous-time solution to the Wiener-Hopf equation and the discrete-time solution to the normal equation are called Wiener filters [27].

To derive the Wiener filter solution for the linear equalizer shown in Fig. 2.2, we adopt the following vector notation. We represent the complex filter tap weights and the filter inputs in the tap delay line at time k using the following length L vectors

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_L]^T, \qquad \boldsymbol{u}(k) = [u_k, u_{k-1}, \ldots, u_{k-L+1}]^T. \tag{2.23}$$

The estimation error is then

$$e_k = d_k - \boldsymbol{w}^H \boldsymbol{u}(k) \tag{2.24}$$

where d_k is the reference signal. Assuming the tap input vector $\boldsymbol{u}(k)$ and the desired response d_k are jointly stationary and d_k has zero mean, the MSE can be written as

$$J(\boldsymbol{w}) = \mathrm{E}\{|e_k|^2\} = \sigma_d^2 - \boldsymbol{p}^H \boldsymbol{w} - \boldsymbol{w}^H \boldsymbol{p} + \boldsymbol{w}^H \boldsymbol{R}\, \boldsymbol{w} \tag{2.25}$$

where $\sigma_d^2 = \mathrm{E}\{|d_k|^2\}$, $\boldsymbol{p} = \mathrm{E}\{\boldsymbol{u}(k)\, d_k^*\}$, and $\boldsymbol{R} = \mathrm{E}\{\boldsymbol{u}(k)\, \boldsymbol{u}^H(k)\}$. To minimize the MSE with respect to the tap weight vector $\boldsymbol{w}$, we take the gradient of $J(\boldsymbol{w})$ with respect to $\boldsymbol{w}^*$ and set the result equal to zero, yielding the normal equation

$$\nabla_{\boldsymbol{w}^*} J(\boldsymbol{w}) = -\boldsymbol{p} + \boldsymbol{R}\, \boldsymbol{w} = \boldsymbol{0}\,. \tag{2.26}$$

Thus, the optimal Wiener filter tap weight vector is

$$\boldsymbol{w}_o = \boldsymbol{R}^{-1} \boldsymbol{p} \tag{2.27}$$

where $\boldsymbol{R}$ is the autocorrelation matrix of the filter input vector, $\boldsymbol{p}$ is the cross-correlation between the filter input vector and the desired response, and $(\cdot)^{-1}$ denotes the matrix inverse operation. Substituting $\boldsymbol{w}_o$ back into (2.25), we finally find the minimum MSE (MMSE)

$$J(\boldsymbol{w}_o) = \sigma_d^2 - \boldsymbol{p}^H \boldsymbol{R}^{-1} \boldsymbol{p}\,. \tag{2.28}$$

2.3.2 Method of Steepest Descent

Instead of performing the computationally complex matrix inversion of $\boldsymbol{R}$, a simple recursive technique called the Method of Steepest Descent (SD) can be used to find the global minimum point of the quadratic multi-dimensional MSE surface $J(\boldsymbol{w})$, corresponding to the optimal tap weight vector $\boldsymbol{w}_o$. The Method of SD uses the exact gradient vector $\nabla J(\boldsymbol{w})$ to converge recursively from some initial set of tap weights

to $\boldsymbol{w}_o$. It requires knowledge of the two ensemble-average quantities, matrix $\boldsymbol{R}$ and vector $\boldsymbol{p}$.

The Method of SD is outlined as follows. First, an initial guess $\boldsymbol{w}(0)$ of the optimum equalizer tap weights vector is formed. Usually, in the absence of prior knowledge of the channel and signal statistics, the initial guess is the zero vector. This guess is then improved by calculating the exact gradient vector $\nabla J(\boldsymbol{w})$, which points in the direction of the greatest rate of change. A scaled version of this gradient vector is subtracted from the previous tap weight vector to update the tap weights. Since the bottom of the convex error performance surface lies in the direction opposite to that of the greatest rate of change. This process is applied iteratively to make the tap weights approach their optimum values [27]. Mathematically, the Method of SD is described by

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \mu \nabla J(\boldsymbol{w}) \tag{2.29}$$

where μ is a step-size parameter. Substituting (2.26) into (2.29), the linear transversal filter tap update equation has the recursive form

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \mu \left[\boldsymbol{p} - \boldsymbol{R}\,\boldsymbol{w}(k)\right]. \tag{2.30}$$

The SD algorithm is simple to implement and converges quickly to the optimum filter coefficients $\boldsymbol{w}_o$, if the step-size μ is chosen appropriately [27]. As we will show in Sec. 2.4, the SD algorithm has a well defined MSE learning curve, consisting of a sum of decaying exponentials. Unfortunately, like the Wiener filter, the SD algorithm requires an exact knowledge of the channel and signal statistics represented by $\boldsymbol{R}$ and $\boldsymbol{p}$. In practice, these quantities must be estimated for the filter to be truly adaptive.

2.3.3 Least Mean Square Algorithm

The well-known adaptive LMS algorithm was first developed by Widrow and Hoff in 1960 [28]. The algorithm is easily derived from the SD algorithm. To bypass the requirement of prior knowledge of the channel and signal statistics, an instantaneous or *stochastic* estimate of the gradient is used in place of the true gradient. This leads

Table 2.3 Computational Complexity of LMS Algorithm

Operation	Complex Multiplies	Complex Adds
Equalizer Output	L	$L-1$
Error & Weight Update	$L+1/2$	$L+1$
Total	$2L+1/2$	$2L$

to a computationally simple recursion for updating the filter tap weights.

Removing the expectation from the left side of (2.26) and using the definition of e_k (2.24), we replace the true gradient $\nabla J(k)$ with the instantaneous estimate

$$\widehat{\nabla} J(k) = -e_k^* \, \boldsymbol{u}(k) \, . \tag{2.31}$$

Note that for a stationary random process this gradient estimate is unbiased because its expected value is exactly the same as the actual gradient vector (2.26). Applying the instantaneous estimate to the Method of SD, the LMS algorithm is then

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \mu \, e_k^* \, \boldsymbol{u}(k) \tag{2.32}$$

with the step-size satisfying $0 < \mu < \frac{1}{\lambda_{\max}}$ where $\lambda_{\max}$ is the largest eigenvalue of the autocorrelation matrix $\boldsymbol{R}$. The LMS algorithm consists of two basic processes [29]:

1. an adaptive process involving the automatic adjustment of the filter tap weights
2. a filter process consisting of:
   (a) calculating the output of the filter based on the filter inputs
   (b) generating an estimation error by comparing this output to a desired response

These two processes work in conjunction to form a feedback loop [27].

The LMS algorithm is very computationally efficient. For each iteration it requires the computations summarized in Table 2.3. Table 2.3 displays computations in the form of *complex* operations, since most of the computations involve multiplying/adding two complex operands. However, in some cases one of the operands is a *real* number (e.g., μ). This halves the complexity of the corresponding multiply/add. Such special

cases are represented as "1/2" of their complex operation counterparts. Details of the complex and real operations are given later in Sec. 3.4.3. Generally, the convergence rate of the LMS algorithm is slow, but it is the simplest and most widely implemented adaptive equalization algorithm.

2.3.4 Normalized Least Mean Square Algorithm

Unfortunately, to choose the appropriate step-size, the LMS algorithm requires *a priori* knowledge of the input power level [30]. Also, fixing the step-size may cause unnecessarily slow convergence in a time-varying channel. To accelerate the equalizer convergence, the Normalized LMS (NLMS) algorithm was proposed in [31, 32]. Modifying the LMS update equation, the tap-weight adjustment for the NLMS Algorithm is "normalized" with respect to the squared Euclidean norm of the tap-input vector $\boldsymbol{u}(k)$. This increases the robustness of the NLMS algorithm against *gradient noise amplification*, which is a problem for the LMS algorithm when $\boldsymbol{u}(k)$ is large and the tap-weight adjustment is directly proportional to $\boldsymbol{u}(k)$.

Like the LMS algorithm, the NLMS algorithm can be used to adapt the transversal filter shown in Fig. 2.2. In this case the filter tap-weights are updated using

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \frac{\mu_o}{\|\boldsymbol{u}(k)\|^2 + \delta}\, e_k^*\, \boldsymbol{u}(k) \tag{2.33}$$

where $0 < \mu_o < 2$ is the adaptation constant and denominator constant $0 < \delta \ll \sigma_u^2$ prevents a division-by-zero from occurring. We see that the error term is normalized with respect to the squared Euclidean norm of the tap-input vector $\boldsymbol{u}(k)$. This recursion can be derived using the *principle of minimal disturbance* [27].

Comparing the NLMS tap-update recursion of (2.33) with its LMS counterpart (2.32), we make the following observations:

- The adaptation constant μ_o in the NLMS algorithm is "dimensionless", whereas the step-size parameter μ in the LMS algorithm has the dimensions of "inverse power".

- The NLMS filter may be viewed as an LMS filter with a "time-varying" step-size

Table 2.4 Computational Complexity of NLMS Algorithm

Operation	Real Divides	Complex Multiplies	Complex Adds
Equalizer Output		L	$L-1$
Norm Update		$1/2$	1
Error & Weight Update	1	$L+1/2$	$L+3/2$
Total	1	$2L+1$	$2L+3/2$

parameter if we set set $\mu(k) = \frac{\mu_o}{\|\boldsymbol{u}(k)\|^2}$.

- Because of this adaptive step-size, the NLMS algorithm potentially converges faster than the LMS algorithm [27].

The cost of faster convergence is the increased complexity of the NLMS algorithm relative to the LMS algorithm. For each iteration it requires the computations summarized in Table 2.4. Practically, dividers are much harder to implement than multipliers. Depending on the application, the effort of building a divider may out-weigh the convergence rate advantage the NLMS algorithm has over the LMS algorithm.

2.3.5 Recursive Least Squares Algorithm

The weakness of LMS-type algorithms, benchmarked by the Wiener filter, is the strong dependence of their convergence rates on the eigenvalue distribution of $\boldsymbol{R}$. The Recursive Least Squares (RLS) algorithm uses an alternative optimization criterion. It recursively calculates the filter tap weights which provide a *least squares fit* of the received observations to the original transmitted symbols. It determines the fixed tap weights which minimize the squared error over a particular observation period.

The cost function to be minimized can be expressed as

$$J_{\mathrm{LS}}(k) = \sum_{i=0}^{k} \lambda^{k-i} |\widetilde{e}_i|^2 + \delta\, \lambda^k \|\boldsymbol{w}(k)\|^2 \tag{2.34}$$

where λ is a "forgetting factor" which ensures that observations in the distant past are "forgotten" (λ is a positive constant close to but less than unity). The error has the form

$$\widetilde{e}_i = d_i - \boldsymbol{w}^H(k)\, \boldsymbol{u}(i) \tag{2.35}$$

where $\boldsymbol{u}(i)$ is the tap-input vector at time i and $\boldsymbol{w}(k)$ is the tap-input vector at time k. Note that the tap weights for the transversal filter remain fixed during the observation interval $1 \leq i \leq k$ for which the cost function $J_{\mathrm{LS}}(k)$ is defined. The first component of (2.34) is the exponentially-weighted "sum of error squares", whereas the second component is a "regularizing term" which smoothes or regularizes the solution to the otherwise ill-posed recursive least-squares problem [27]. The small positive real number δ is called the regularizing parameter.

We can show that the optimum tap-weight vector $\boldsymbol{w}(k)$, for which the cost function of (2.34) attains its minimum value, satisfies the least-squares version of the normal equation (2.26), written as

$$\boldsymbol{\Phi}(k)\,\boldsymbol{w}(k) = \boldsymbol{\zeta}(k)\,. \tag{2.36}$$

Here, the time-averaged autocorrelation matrix of the tap-input vector $\boldsymbol{u}(i)$ has the form

$$\boldsymbol{\Phi}(k) = \sum_{i=1}^{k} \lambda^{k-i}\,\boldsymbol{u}(i)\,\boldsymbol{u}^{H}(i) + \delta\,\lambda^{k}\boldsymbol{I}_{L} \tag{2.37}$$

where $\boldsymbol{I}_L$ is the $L \times L$ identity matrix. Similarly, the $L \times 1$ time-averaged cross-correlation vector between the tap inputs of the transversal filter and the desired response is

$$\boldsymbol{\zeta}(k) = \sum_{i=1}^{k} \lambda^{k-i}\,\boldsymbol{u}(i)\,d_i^{*}\,. \tag{2.38}$$

To compute recursively the solution to (2.36), we can isolate the term corresponding to $i = k$ on the right-hand side of (2.37) from the rest of the summation and write

$$\boldsymbol{\Phi}(k) = \lambda\left[\sum_{i=1}^{k-1} \lambda^{k-1-i}\boldsymbol{u}(i)\,\boldsymbol{u}^{H}(i) + \delta\,\lambda^{k-1}\boldsymbol{I}_{L}\right] + \boldsymbol{u}(k)\,\boldsymbol{u}^{H}(k)\,. \tag{2.39}$$

We observe the expression inside the brackets on the right-hand side of (2.39) is equivalent to the correlation matrix $\boldsymbol{\Phi}(k-1)$. We can derive $\boldsymbol{\zeta}(k-1)$ in a similar fashion

from (2.38). Hence, $\boldsymbol{\Phi}(k)$ and $\boldsymbol{\zeta}(k)$ can be computed recursively using

$$\begin{aligned}\boldsymbol{\Phi}(k) &= \lambda\,\boldsymbol{\Phi}(k-1) + \boldsymbol{u}(k)\,\boldsymbol{u}^H(k)\\ \boldsymbol{\zeta}(k) &= \lambda\,\boldsymbol{\zeta}(k-1) + \boldsymbol{u}(k)\,d_k^*\,.\end{aligned} \tag{2.40}$$

To obtain the least-squares estimate $\boldsymbol{w}(k)$ satisfying (2.36), we want to avoid computing the inverse of $\boldsymbol{\Phi}(k)$. To do this, we invoke the well-known Matrix Inversion Lemma for positive definite symmetric matrices $\boldsymbol{A}$, $\boldsymbol{B}$ with $\boldsymbol{A} = \boldsymbol{B}^{-1} + \boldsymbol{C}\,\boldsymbol{D}^{-1}\boldsymbol{C}^H$. In this case, it can be shown that

$$\boldsymbol{A}^{-1} = \boldsymbol{B} - \boldsymbol{B}\,\boldsymbol{C}(\boldsymbol{D} + \boldsymbol{C}^H\boldsymbol{B}\,\boldsymbol{C})^{-1}\boldsymbol{C}^H\boldsymbol{B}\,.$$

Using the definitions $\boldsymbol{A} = \boldsymbol{\Phi}(k)$, $\boldsymbol{B}^{-1} = \lambda\,\boldsymbol{\Phi}(k-1)$, $\boldsymbol{C} = \boldsymbol{u}(k)$ and $\boldsymbol{D} = 1$, we obtain the new recursion

$$\boldsymbol{\Phi}^{-1}(k) = \lambda^{-1}\boldsymbol{\Phi}^{-1}(k-1) - \frac{\lambda^{-2}\,\boldsymbol{\Phi}^{-1}(k-1)\,\boldsymbol{u}(k)\boldsymbol{u}^H(k)\,\boldsymbol{\Phi}^{-1}(k-1)}{1 + \lambda^{-1}\,\boldsymbol{u}^H(k)\,\boldsymbol{\Phi}^{-1}(k-1)\,\boldsymbol{u}(k)}\,. \tag{2.41}$$

To simplify the notation, setting

$$\boldsymbol{P}(k) = \boldsymbol{\Phi}^{-1}(k) \tag{2.42}$$

$$\boldsymbol{k}(k) = \lambda^{-1}\boldsymbol{P}(k-1)\,\boldsymbol{u}(k)\left[1 + \lambda^{-1}\,\boldsymbol{u}^H(k)\,\boldsymbol{P}(k-1)\,\boldsymbol{u}(k)\right]^{-1}, \tag{2.43}$$

we may rewrite recursion (2.41) as

$$\boldsymbol{P}(k) = \lambda^{-1}\boldsymbol{P}(k-1) - \lambda^{-1}\boldsymbol{k}(k)\,\boldsymbol{u}^H(k)\,\boldsymbol{P}(k-1)\,. \tag{2.44}$$

The $L \times L$ matrix $\boldsymbol{P}(k)$ is referred to as the *inverse correlation matrix* while the $L \times 1$ vector $\boldsymbol{k}(k)$ is the referred to as the *gain vector*. Equation (2.44) is the *Riccati equation* for the RLS algorithm. Rearranging (2.43), we can write

$$\begin{aligned}\boldsymbol{k}(k) &= \lambda^{-1}\boldsymbol{P}(k-1)\,\boldsymbol{u}(k) - \lambda^{-1}\,\boldsymbol{k}(k)\,\boldsymbol{u}^H(k)\,\boldsymbol{P}(k-1)\,\boldsymbol{u}(k)\\ &= \left[\lambda^{-1}\boldsymbol{P}(k-1) - \lambda^{-1}\,\boldsymbol{k}(k)\,\boldsymbol{u}^H(k)\,\boldsymbol{P}(k-1)\right]\,\boldsymbol{u}(k)\,.\end{aligned} \tag{2.45}$$

We observe that the expression inside the brackets on the right-hand side of the second line of (2.45) equals $\boldsymbol{P}(k)$ from (2.44). Hence, we may simplify (2.45) to

$$\boldsymbol{k}(k) = \boldsymbol{P}(k)\,\boldsymbol{u}(k)\,. \tag{2.46}$$

Returning to the normal equation (2.36), we can express the tap-weight vector in terms of the inverse correlation matrix and then use (2.40) in place of $\boldsymbol{\zeta}(k)$, producing

$$\begin{aligned}\boldsymbol{w}(k) &= \boldsymbol{\Phi}^{-1}(k)\,\boldsymbol{\zeta}(k) = \boldsymbol{P}(k)\,\boldsymbol{\zeta}(k)\\ &= \lambda\,\boldsymbol{P}(k)\,\boldsymbol{\zeta}(k-1) + \boldsymbol{P}(k)\,\boldsymbol{u}(k)\,d_k^*\,.\end{aligned} \tag{2.47}$$

Substituting (2.44) for $\boldsymbol{P}(k)$ on the right side of (2.47) yields

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) - \boldsymbol{k}(k)\,\boldsymbol{u}^H(k)\,\boldsymbol{w}(k-1) + \boldsymbol{P}(k)\,\boldsymbol{u}(k)\,d_k^*\,. \tag{2.48}$$

Finally, using (2.46), the RLS tap-weight update equation has the final form

$$\begin{aligned}\boldsymbol{w}(k) &= \boldsymbol{w}(k-1) - \boldsymbol{k}(k)\left[d_k^* - \boldsymbol{u}(k)^H\,\boldsymbol{w}(k-1)\right]\\ &= \boldsymbol{w}(k-1) - \boldsymbol{k}(k)\,\epsilon_k^*\end{aligned} \tag{2.49}$$

with *a priori* estimation error

$$\epsilon_k = d_k - \boldsymbol{w}^H(k-1)\,\boldsymbol{u}(k)\,. \tag{2.50}$$

The RLS algorithm is summarized in Table 2.5.

Again, the cost of faster convergence is the increased complexity of the RLS algorithm relative to the LMS and NLMS algorithms. For each iteration the RLS algorithm requires the computations summarized in Table 2.6. Care has to be taken to make sure that the real-time implementation of the RLS algorithm is feasible in a given application. Having reviewed the classical adaptive equalization algorithms, we now consider the classical equalization architecture, the DFE.

Table 2.5 Summary of Conventional RLS Algorithm

Initial conditions: $\boldsymbol{w}(0) = \mathbf{0}$ $\boldsymbol{P}(0) = \delta^{-1}\boldsymbol{I}_L$ where δ is a small positive constant for high SNR
To update taps at each iteration, compute $\boldsymbol{\pi}(k) = \boldsymbol{P}(k-1)\,\boldsymbol{u}(k)$ $\boldsymbol{k}(k) = \frac{\boldsymbol{\pi}(k)}{\lambda + \boldsymbol{u}^H(k)\,\boldsymbol{\pi}(k)}$ $\epsilon_k = d_k - \boldsymbol{w}^H(k-1)\,\boldsymbol{u}(k)$ $\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \boldsymbol{k}(k)\,\epsilon_k^*$ $\boldsymbol{P}(k) = \lambda^{-1}\boldsymbol{P}(k-1) - \lambda^{-1}\boldsymbol{k}(k)\,\boldsymbol{u}^H(k)\,\boldsymbol{P}(k-1)$

Table 2.6 Computational Complexity of RLS Algorithm

Operation	Real Divides	Complex Multiplies	Complex Adds
$\boldsymbol{\pi}(k)$ Update		L^2	$L(L-1)$
$\boldsymbol{k}(k)$ Update	1	$3L/2$	$L - 1/2$
ϵ_k Update		L	L
$\boldsymbol{w}(k)$ Update		L	L
$\boldsymbol{P}(k)$ Update		$5L^2/2$	$L(2L-1)$
Total	1	$7/2\,(L^2 + L)$	$3L^2 + L - 1/2$

2.3.6 Decision Feedback Equalizer

Thus far, we have only considered the linear transversal filter structure shown in Fig 2.2. But, by adding an additional linear feedback filter to the linear equalizer after the decision device, it is well known that the new "decision feedback" structure shown in Fig. 2.3 is even better at mitigating the ISI than a linear equalizer [27]. By using the information in the previously detected symbols to help detect the current symbol, this relatively simple nonlinear decision feedback equalizer (DFE) achieves significant performance gains. The feedback section actually subtracts out the portion of the ISI produced by previously detected symbols from the estimates of future samples.

As with the linear equalizer, for a DFE we can derive the optimal Wiener tap weights which minimize the MSE [33]. First, we let the feedforward and feedback filters have the respective tap weights

$$\boldsymbol{f} = [f_1, f_2, \ldots, f_{N_f}]^T, \qquad \boldsymbol{b} = [b_1, b_2, \ldots, b_{N_b}]^T. \tag{2.51}$$

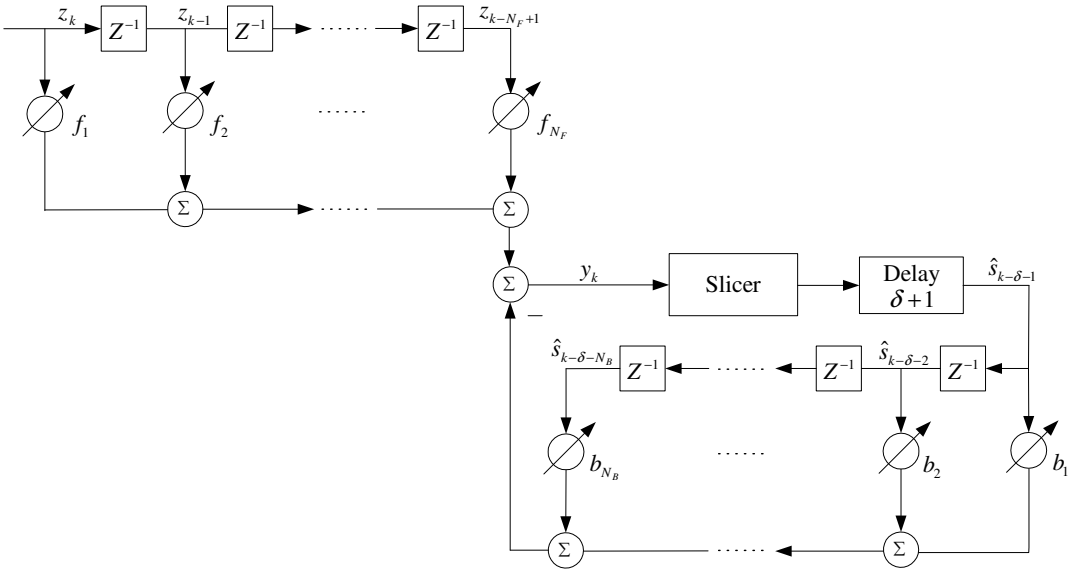

Figure 2.3 A generic DFE architecture

Using this vector notation, we can reduce Fig. 2.3 to Fig. 2.4. During equalizer train-

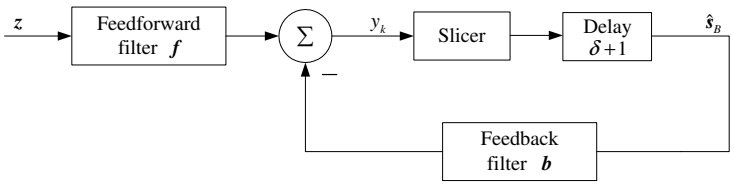

Figure 2.4 Block diagram of DFE

ing, we assume that the actual transmitted symbols s_k corresponding to the equalizer outputs are fed back into the feedback filter. Thus, for an equalizer that is trying to recover symbol $s_{k-\delta}$ with propagation delay δ, the feedback filter data vector is $\boldsymbol{s}_B = [s_{k-\delta-1}, s_{k-\delta-2}, \ldots, s_{k-\delta-N_b}]^T$.

The minimization of the MSE can therefore be cast into the vector form

$$\min_{\boldsymbol{f},\boldsymbol{b}} \mathrm{E}\{|s_{k-\delta} - \boldsymbol{f}^H \boldsymbol{z} + \boldsymbol{b}^H \boldsymbol{s}_B|^2\} = \min_{\boldsymbol{w}} \mathrm{E}\{|s_{k-\delta} - \boldsymbol{w}^H \boldsymbol{v}|^2\} = \min_{\boldsymbol{w}} \mathrm{E}\{|e_k|^2\} \qquad (2.52)$$

where the super-weight and super-observation vectors are

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{b} \end{bmatrix}, \quad \boldsymbol{v} = \begin{bmatrix} \boldsymbol{z} \\ -\boldsymbol{s}_B \end{bmatrix}, \tag{2.53}$$

the feedforward filter input vector is $\boldsymbol{z} = [z_k, z_{k-1}, \ldots, z_{k-N_f+1}]^T$, and the estimation error for the DFE is

$$e_k = s_{k-\delta} - \boldsymbol{w}^H \boldsymbol{v}\,. \tag{2.54}$$

Expanding (2.52) and using the linearity of the expectation, we can rewrite the MSE as

$$J(\boldsymbol{w}) = \sigma_s^2 - \boldsymbol{w}^H \mathrm{E}\{\boldsymbol{v}\, s_{k-\delta}^*\} - \mathrm{E}\{s_{k-\delta}\, \boldsymbol{v}^H\}\, \boldsymbol{w} + \boldsymbol{w}^H \mathrm{E}\{\boldsymbol{v}\, \boldsymbol{v}^H\}\, \boldsymbol{w} \tag{2.55}$$

where $\sigma_s^2 = \mathrm{E}\{|s_k|^2\}$. To find the Wiener filter tap coefficients, we again take the gradient with respect to $\boldsymbol{w}^*$ and set it equal to zero, yielding

$$\nabla_{\boldsymbol{w}^*} J(\boldsymbol{w}) = -\mathrm{E}\{\boldsymbol{v}\, s_{k-\delta}^*\} + \mathrm{E}\{\boldsymbol{v}\, \boldsymbol{v}^H\}\, \boldsymbol{w}_{\mathrm{opt},\,\delta} = \boldsymbol{0}\,. \tag{2.56}$$

Solving for $\boldsymbol{w}_{\mathrm{opt},\,\delta}$, we find the optimal coefficients corresponding to delay δ are

$$\boldsymbol{w}_{\mathrm{opt},\,\delta} = \left[\mathrm{E}\{\boldsymbol{v}\, \boldsymbol{v}^H\}\right]^{-1} \mathrm{E}\{\boldsymbol{v}\, s_{k-\delta}^*\} \stackrel{\triangle}{=} \boldsymbol{R}_\delta^{-1}\, \boldsymbol{p}_\delta\,. \tag{2.57}$$

Equation (2.57) is the combined feedforward and feedback Weiner solution. Substituting (2.53) into (2.57), autocorrelation matrix $\boldsymbol{R}_\delta$ and crosscorrelation vector $\boldsymbol{p}_\delta$ can be written

$$\boldsymbol{R}_\delta = \begin{bmatrix} \mathrm{E}\{\boldsymbol{z}\, \boldsymbol{z}^H\} & -\mathrm{E}\{\boldsymbol{z}\, \boldsymbol{s}_B^H\} \\ -\mathrm{E}\{\boldsymbol{s}_B\, \boldsymbol{z}^H\} & \mathrm{E}\{\boldsymbol{s}_B\, \boldsymbol{s}_B^H\} \end{bmatrix}, \qquad \boldsymbol{p}_\delta = \begin{bmatrix} \mathrm{E}\{\boldsymbol{z}\, s_{k-\delta}^*\} \\ \boldsymbol{0}_{N_b \times 1} \end{bmatrix} \tag{2.58}$$

assuming i.i.d. symbols $\{s_k\}$ and delay $\delta > 0$. Again, analogous versions of the Method of SD (2.30) and the LMS, NLMS and RLS algorithms can be formulated for the DFE. One final note is that a fractionally-spaced version of the DFE can be formed by using

a sample-spaced feedforward filter and a symbol-spaced feedback filter.

2.4 ANALYSIS OF ADAPTIVE EQUALIZER CONVERGENCE BEHAVIOR

Returning to the LMS algorithm of Sec. 2.3.3, although it is very simple to implement, its mathematical analysis is quite complicated because of its feedback structure and nonlinearity. Despite extensive efforts to analyze the LMS algorithm, a direct mathematical analysis of its stability and steady-state performance has generally eluded researchers [27]. The Method of SD, which uses the exact gradient, gives an overly optimistic lower bound on the MSE convergence trajectory over time. Thus, we consider two alternative statistical methods to compute the MSE trajectory, the Independence Method and the Small Step-size Method, which better account for the noisy gradient estimate in the LMS recursion [27]. In this thesis we limit our dynamic convergence analysis to the LMS algorithm. But, similar calculations for the NLMS and RLS algorithms can be performed using the statistical methods discussed in [30] and [27, Sec. 9.7], respectively.

For the rest of this section, we return to the notation of Sec. 2.3.1 for the linear equalizer of Fig. 2.2, but the notation and analysis can be easily extended to the DFE of Sec. 2.3.6. For statistical analysis of the LMS algorithm, it is more convenient to work with the weight error vector than with the tap weight vector itself. Here, we denote the weight error vector by

$$\boldsymbol{\varepsilon}(k) = \boldsymbol{w}_o - \boldsymbol{w}(k) \tag{2.59}$$

where $\boldsymbol{w}_o$ is the optimum Wiener tap-weight vector and $\boldsymbol{w}(k)$ is the estimate produced by the LMS algorithm at iteration k. Therefore, we may rewrite the LMS algorithm of (2.32) in terms of the weight error vector $\boldsymbol{\varepsilon}(k)$ [34]:

$$\boldsymbol{\varepsilon}(k+1) = [\boldsymbol{I} - \mu\,\boldsymbol{u}(k)\,\boldsymbol{u}^H(k)]\,\boldsymbol{\varepsilon}(k) - \mu\,\boldsymbol{u}(k)\,e_{ok}^* \tag{2.60}$$

where $\boldsymbol{I}$ is the identity matrix and

$$e_{ok} = d_k - \boldsymbol{w}_o^H \boldsymbol{u}(k) \tag{2.61}$$

is the optimum Wiener filter's estimation error. To adapt this notation for the DFE, we can simply replace d_k by s_k, $\boldsymbol{u}(k)$ by $\boldsymbol{v}(k)$, and error (2.61) by $e_{ok} = s_k - \boldsymbol{w}_o^H \boldsymbol{v}(k)$.

2.4.1 Independence Method

Our first approach to analyzing the statistical behavior of the LMS algorithm is called the Independence Method and is based on the following assumptions and approximations [29]:

(I-1) The tap-input vectors $\boldsymbol{u}(1)$, $\boldsymbol{u}(2)$,..., $\boldsymbol{u}(k)$ are statistically independent of each other.

(I-2) At time k, the tap-input vector $\boldsymbol{u}(k)$ is statistically dependent on the desired response d_k but independent of all previous samples of the desired response: d_1, d_2,..., d_{k-1}.

(I-3) At time k, the tap weight error vector $\boldsymbol{\varepsilon}(k)$ is independent of both $\boldsymbol{u}(k)$ and d_k.

(I-4) The tap-input vector $\boldsymbol{u}(k)$ and the desired response d_k are mutually Gaussian-distributed random variables.

Note that these assumptions can be drastic in some scenarios (e.g., **(I-4)** will not be true for an ISI channel). Detailed justifications for each of these assumptions are discussed in [34].

To determine the MSE trajectory as the LMS algorithm adapts over time, we need to compute the first two moments of the weight-error vector $\boldsymbol{\varepsilon}(k)$. We start with the first moment by taking the expectation of (2.60). Applying Assumption **(I-3)** and invoking the orthogonality property of the optimal weight vector $\boldsymbol{w}_o$,

$$\mathrm{E}\{\boldsymbol{u}(k)\, e_{ok}^*\} = \boldsymbol{p} - \boldsymbol{R}\,\boldsymbol{w}_o = \boldsymbol{0}\,, \tag{2.62}$$

we find the recursion

$$\mathrm{E}\{\boldsymbol{\varepsilon}(k+1)\} = (\boldsymbol{I} - \mu\,\boldsymbol{R})\,\mathrm{E}\{\boldsymbol{\varepsilon}(k)\}\,. \tag{2.63}$$

Assuming that all of the eigenvalues of $(\boldsymbol{I} - \mu\boldsymbol{R})$ are strictly less than one, this recursion describes a stable feedback loop [29], and $\mathrm{E}\{\boldsymbol{\varepsilon}(k)\} \longrightarrow \mathbf{0}$ as $k \longrightarrow \infty$, yielding $\mathrm{E}\{\boldsymbol{w}(k)\} \longrightarrow \boldsymbol{w}_o$ as $k \longrightarrow \infty$. Under this condition, the LMS algorithm is *convergent in the mean*.

Similarly, we can derive the time evolution of the weight-error correlation matrix

$$\boldsymbol{K}(k) = \mathrm{E}\{\boldsymbol{\varepsilon}(k)\boldsymbol{\varepsilon}(k)^H\}\,. \tag{2.64}$$

Using the LMS update equation (2.60) and applying Assumptions **(I-1)**–**(I-4)** along with the Gaussian moment theorem [34], we can show that

$$\begin{aligned}\boldsymbol{K}(k+1) = \boldsymbol{K}(k) &- \mu\,\boldsymbol{R}\,\boldsymbol{K}(k) - \mu\,\boldsymbol{K}(k)\boldsymbol{R} + \mu^2\boldsymbol{R}\,\boldsymbol{K}(k)\boldsymbol{R} \\ &+ \mu^2\boldsymbol{R}\,\mathrm{Tr}[\boldsymbol{R}\,\boldsymbol{K}(k)] + \mu^2 J_{\min}\,\boldsymbol{R}\,.\end{aligned} \tag{2.65}$$

Finally, the evolution of the MSE over time is obtained according to:

$$J(k) = J_{\min} + \mathrm{Tr}[\boldsymbol{R}\,\boldsymbol{K}(k)] \tag{2.66}$$

where MMSE $J_{\min} = \sigma_d^2 - \boldsymbol{p}^H\boldsymbol{R}^{-1}\boldsymbol{p}$, and $\mathrm{Tr}[\boldsymbol{R}\,\boldsymbol{K}(k)]$ converges to the steady-state excess MSE. To apply the Independence Method to the DFE, we only need to replace $\boldsymbol{p}$ and $\boldsymbol{R}$ by $\boldsymbol{p}_\delta$ and $\boldsymbol{R}_\delta$ of (2.58).

As a first simple equalization scenario we study the LMS algorithm convergence behavior for a $T/5$-spaced DFE in a BPSK transceiver sending data over the impulse-like channel with response shown in Fig. 2.5. The impulse channel is a real zero-pole flat fading channel with a delay spread of exactly one symbol (5 samples). Figure 2.6 shows the average MSE trajectory for the LMS algorithm compared with the MSE trajectories generated using the Method of SD, the Independence Method and the Small Step-size Method. The Monte Carlo simulations are averaged over 100 trials. The DFE has 5

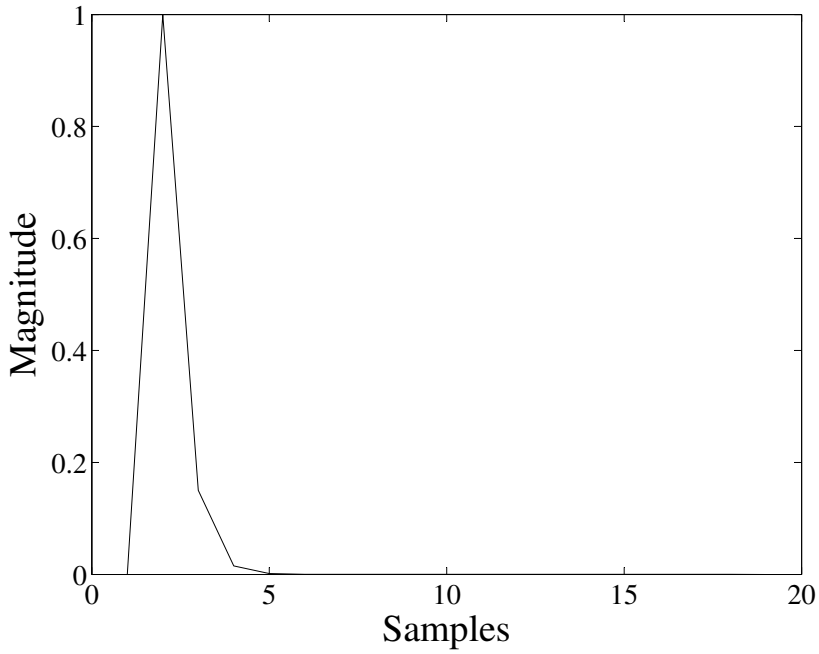

Figure 2.5 Impulse-like channel response

feedforward taps, 2 feedback taps, step-size $\mu = 0.05$ and all-zero initial tap-weights.

For the impulse-like (minimum ISI) channel we see that the Independence Method most closely follows the LMS trajectory, lending credence to the theoretical assumptions **(I-1)**–**(I-4)**. However, for the frequency selective TR-STBC channels we consider in this thesis the tap input vectors $\boldsymbol{u}(1), \ldots, \boldsymbol{u}(k)$ are in fact highly statistically dependent. This can lead to inaccuracy in the independence assumption-based analysis. As an alternative, we now review Butterweck's small step-size theory based on the direct averaging method of Kushner [35].

2.4.2 Small Step-size Theory

We note that (2.60) is a stochastic difference equation in the weight error vector $\boldsymbol{\varepsilon}(k)$ for a system matrix equal to $[\boldsymbol{I} - \mu\, \boldsymbol{u}(k)\boldsymbol{u}^H(k)]$. Under the assumption of a small step-size μ, Kushner [35] demonstrated that the solution for (2.60) is close to the solution for another stochastic difference equation whose system matrix is equal to the ensemble average

$$\mathrm{E}\{\boldsymbol{I} - \mu\, \boldsymbol{u}(k)\boldsymbol{u}^H(k)\} = \boldsymbol{I} - \mu\, \boldsymbol{R}\,. \tag{2.67}$$

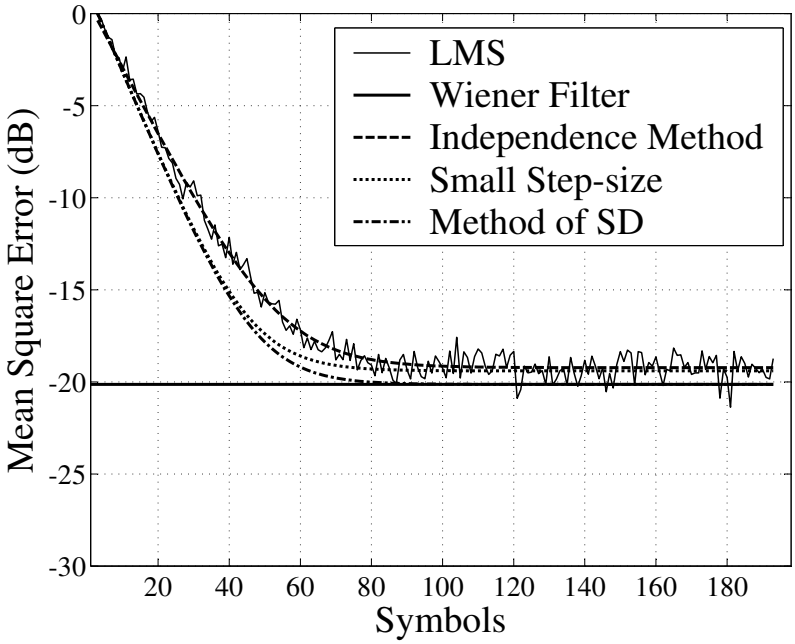

Figure 2.6 MSE convergence curves for $T/5$-spaced DFE over impulse-like channel

The stochastic difference equation (2.60) can be replaced by the alternative stochastic difference equation

$$\boldsymbol{\varepsilon}_0(k+1) = (\boldsymbol{I} - \mu\,\boldsymbol{R})\,\boldsymbol{\varepsilon}_0(k) - \mu\,\boldsymbol{u}(k)\,e_{ok}^* \tag{2.68}$$

which has the same solution as (2.60) for the limiting case of small μ. Butterweck [27] has proposed a "small step-size method", where the solution of (2.68) is used as a starting point to iteratively generate a whole set of solutions for (2.60). (This is why we have replaced $\boldsymbol{\varepsilon}(k)$ by $\boldsymbol{\varepsilon}_0(k)$.)

To analyze (2.68) using the Small Step-size Method, we adopt the following new assumptions [27]:

(S-1) The step-size parameter μ is small, so the LMS filter acts as a low-pass filter with a low cutoff frequency.

(S-2) The physical mechanism for generating d_k is described by a linear multiple regression model that is matched exactly by the Wiener filter; that is,

$$d_k = \boldsymbol{w}_o^H\,\boldsymbol{u}(k) + e_{ok} \tag{2.69}$$

where the irreducible estimation error e_{ok} is a white-noise process statistically independent of the input vector $\boldsymbol{u}(k)$.

(S-3) The input vector $\boldsymbol{u}(k)$ and the desired response d_k are jointly Gaussian.

Under Assumption **(S-1)**, Butterweck showed that that the iteratively generated solutions to the stochastic difference equation are governed by

$$\boldsymbol{\varepsilon}_0(k+1) = (\boldsymbol{I} - \mu\,\boldsymbol{R})\,\boldsymbol{\varepsilon}_0(k) + \boldsymbol{f}_o(k) \tag{2.70}$$

where $\boldsymbol{f}_o(k) = -\mu\,\boldsymbol{u}(k)\,e_{ok}^*$. We can then form the eigendecomposition $\boldsymbol{R} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^H$, where $\boldsymbol{Q}$ is a unitary matrix containing the eigenvectors of $\boldsymbol{R}$ and $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues $\lambda_1, \ldots, \lambda_L$ of $\boldsymbol{R}$. If we define

$$\boldsymbol{v}(k) = \boldsymbol{Q}^H\boldsymbol{\varepsilon}_0(k), \qquad \boldsymbol{\phi}(k) = \boldsymbol{Q}^H\boldsymbol{f}_o(k), \tag{2.71}$$

we may transform (2.70) into the form

$$\boldsymbol{v}(k+1) = (\boldsymbol{I} - \mu\,\boldsymbol{\Lambda})\,\boldsymbol{v}(k) + \boldsymbol{\phi}(k). \tag{2.72}$$

Recursion (2.72) contains L parallel decoupled difference equations or "natural modes" constituting the transient response of the LMS algorithm, where the k-th natural mode is of the form

$$v_m(k+1) = (1 - \mu\,\lambda_m)\,v_m(k) + \phi_m(k), \qquad m = 1, 2, \ldots, L\,. \tag{2.73}$$

It follows that the change in the natural mode v_m from one iteration to the next is

$$\Delta\,v_m(k) = v_m(k+1) - v_m(k) = -\mu\,\lambda_m\,v_m(k) + \phi_m(k) \tag{2.74}$$

which can be split into two parts: a *damping force* $-\mu\,\lambda_m\,v_m(k)$ and a *stochastic force* $\phi_m(k)$. Given that $\phi_m(k)$ is a zero-mean white noise process, we recognize (2.74) as the discrete-time version of the Langevin equation which characterizes Brownian motion [36, 37]. Providing that the step-size parameter of the LMS filter is small, the natural

modes of the filter execute Brownian motion about some fixed values.

We can show that the first and second moments of the natural modes reduce to

$$\mathrm{E}\{v_m(k)\} = v_m(0)(1-\mu\,\lambda_m)^k \tag{2.75}$$

$$\mathrm{E}\{|v_m(k)|^2\} = \frac{\mu\,J_{\min}}{2-\mu\,\lambda_m} + (1-\mu\,\lambda_m)^{2k}\left(|v_m(0)|^2 - \frac{\mu\,J_{\min}}{2-\mu\,\lambda_m}\right) \tag{2.76}$$

with initial condition $\boldsymbol{v}(0) = \boldsymbol{Q}^H[\boldsymbol{w}_o - \boldsymbol{w}(0)]$. Finally, using this Small Step-size Method, the MSE has the final form

$$\begin{aligned} J(k) &= J_{\min} + \sum_{m=1}^{L} \lambda_m\,\mathrm{E}\{|v_m(k)|^2\} \\ &= J_{\min} + \mu\,J_{\min}\sum_{m=1}^{L}\frac{\lambda_m}{2-\mu\,\lambda_m} + \sum_{m=1}^{L}\lambda_m\left[|v_m(0)|^2 - \frac{\mu\,J_{\min}}{2-\mu\,\lambda_m}\right](1-\mu\,\lambda_m)^{2k} \\ &\simeq J_{\min} + \frac{\mu\,J_{\min}}{2}\sum_{m=1}^{L}\lambda_m + \sum_{m=1}^{L}\lambda_m\left[|v_m(0)|^2 - \frac{\mu\,J_{\min}}{2}\right](1-\mu\,\lambda_m)^{2k} \end{aligned} \tag{2.77}$$

where the last approximation holds for small μ. This is naturally satisfied since the earlier analysis leading to (2.77) requires that μ be small. We see that the MSE trajectory has two components. The first two terms on the right side of (2.77) comprise the steady state MSE, whereas the last term contains the dynamic component which decays as a sum of exponentials.

Increasing the effective ISI, we calculate the LMS convergence behavior for the $T/5$-spaced DFE in a BPSK transceiver sending data over the channel with response plotted in Fig. 2.7. This fixed indoor wireless channel impulse response was captured using the STAR platform [11] and is discussed in more detail in Sec. 3.5.1. Figure 2.8 shows the average MSE trajectory for the LMS algorithm compared with the MSE trajectories generated using the Method of SD, the Independence Method and the Small Step-size Method. The Monte Carlo simulations are averaged over 100 trials. The DFE is identical to the one in Sec. 2.4.1 with 5 feedforward taps, 2 feedback taps, step-size $\mu = 0.05$ and all-zero initial tap-weights. Again, in spite of the increased ISI, the Independence Method most closely follows the LMS trajectory.

Because we consider a variety of channel types and characteristics in the next chap-

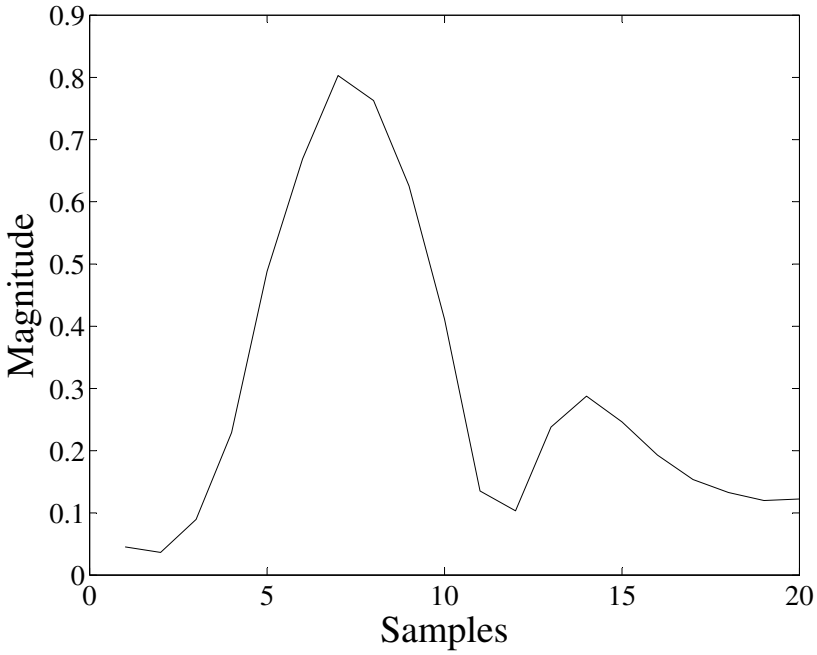

Figure 2.7 Measured fixed indoor wireless channel response

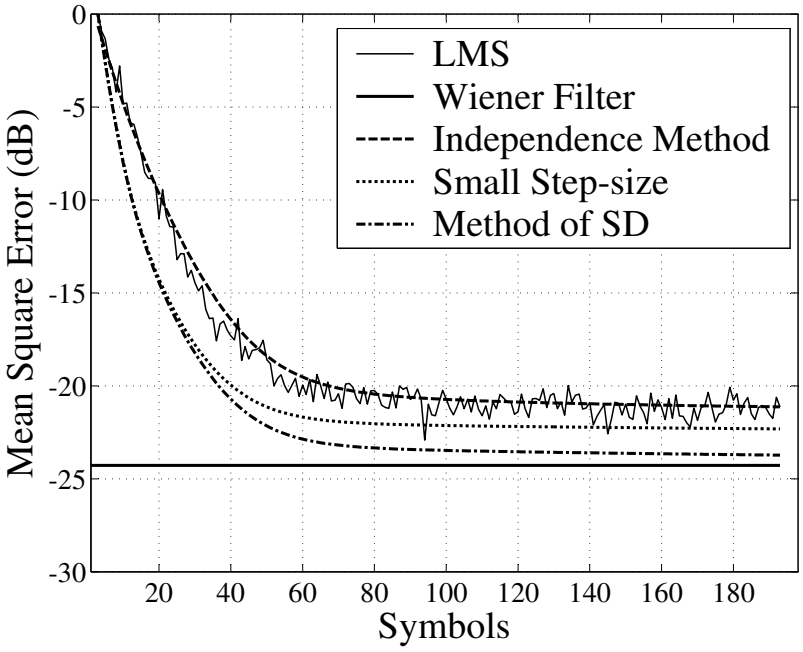

Figure 2.8 MSE convergence curves for $T/5$-spaced DFE over measured fixed indoor wireless channel

ters, despite the superior performance of the Independence Method over the Small Step-size Method for the particular channels in Figs. 2.6 and 2.8, we consider both

statistical methods when we apply our analysis to space time block codes.

Having reviewed space time block coding and adaptive equalization techniques, we are now ready to combine them to form a robust wireless communication system that can be practically implemented.

Chapter 3

DFE STRUCTURES AND ALGORITHMS FOR A 2×1 TR-STBC SYSTEM

In Chapter 2.1 we introduced the 2×1 TR-STBC system and the Single-Input Single-Output (SISO) DFE. Combining these two technologies, in this chapter we analyze the performance of TR-STBC systems with different DFE equalization schemes and adaptive algorithms. We first introduce three alternative DFE architectures for the 2×1 TR-STBC system. Along with conventional uncoupled SISO DFEs and the fully-coupled MIMO DFE, we propose a novel hybrid DFE with new adaptive filtering algorithms which outperforms the two conventional structures. We derive the Wiener filter solutions for all three architectures, as well as analyze their dynamic convergence behaviors when adapted using architecture-specific versions of the LMS algorithm. Since fast convergence of the equalizer taps is required for wireless applications, we also investigate two training method enhancements involving a new prototype block structure and a kick-start scheme using the Method of SD. As an alternative to the LMS algorithm, we also propose new versions of the NLMS and RLS algorithms, which lead to even faster equalizer tap convergence. We then analyze the computational complexity of each of the architectures and algorithms with the goal of real-time implementation. Finally, we review our wireless channel models before using Monte Carlo simulations to compare the equalizer structures and algorithms.

3.1 ALTERNATIVE 2×1 TR-STBC DFE STRUCTURES

Combining the 2×1 TR-STBC system of Sec. 2.2 (Fig. 2.1) with the DFE architecture of Sec. 2.3.6 (Fig. 2.4), we propose three competing DFE structures for ISI mitigation

after the TR-STBC decoder. The first involves two parallel decoupled SISO DFEs shown in Fig. 3.1, which independently mitigate the ISI in the two decoder outputs processes $z_1(t)$ and $z_2(t)$.

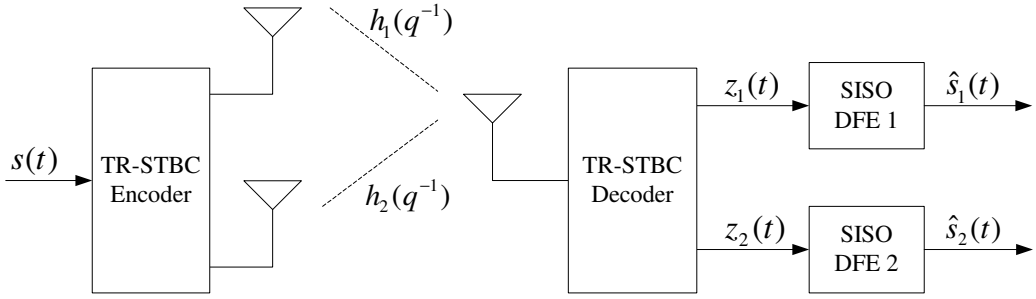

Figure 3.1 2 × 1 TR-STBC system with SISO DFEs

For the case of perfect knowledge of the channel at the receiver, output processes $z_1(t)$ and $z_2(t)$ are theoretically decoupled. However, for the practical case of imperfect channel estimates, the processes in general will be coupled. In this case, joint processing of $z_1(k)$ and $z_2(k)$ could be beneficial and more robust to the loss of orthogonality. Thus, the other conventional DFE structure we consider is the 2 × 2 MIMO DFE shown in Fig. 3.2. Ideally this MIMO DFE structure should help mitigate potential signal leakage between the output processes.

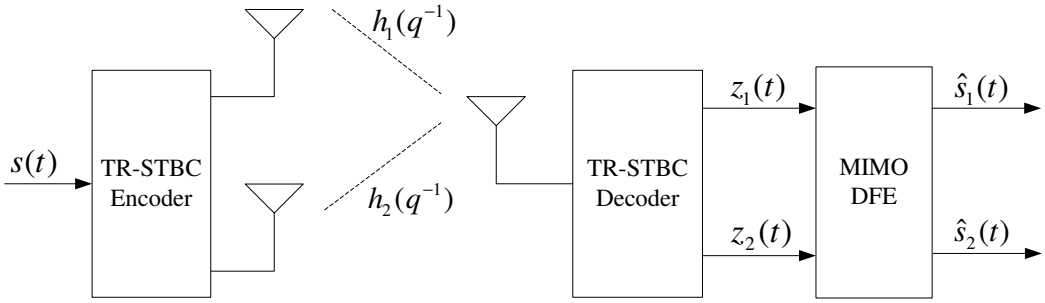

Figure 3.2 2 × 1 TR-STBC system with MIMO DFE

Finally, based on our Wiener filter analysis of the original parallel SISO DFEs, we introduce the novel hybrid DFE shown in Fig. 3.3. Due to the common second-order statistics between $z_1(t)$ and $z_2(t)$, we retain the parallel SISO DFE structure, but con-

strain the tap weights in the two parallel equalizers to be the same. Instead of utilizing the standard LMS algorithm of Sec. 2.3.3, we use a modified LMS algorithm, originally proposed by Gao and Leung [38] for CAP equalization in DSL applications. This algorithm combines the stochastic estimates of the error gradients for both equalizers, leading to a clever exchange of error information between the two parallel SISO DFEs and better equalization performance.

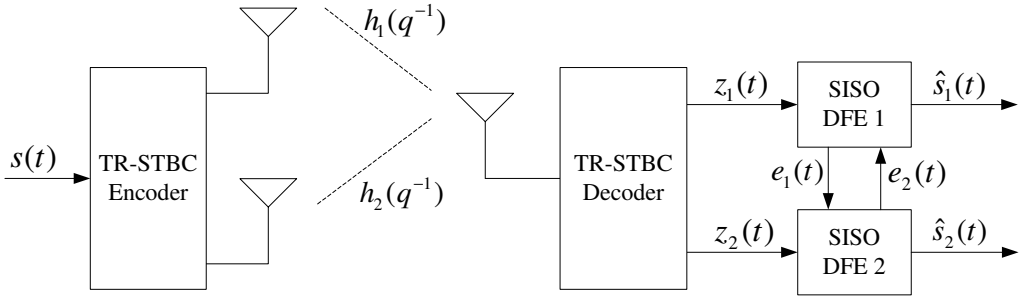

Figure 3.3 2 × 1 TR-STBC system with Hybrid DFE

3.1.1 Parallel SISO DFEs

We first analyze the two decoupled parallel SISO DFEs of Fig. 3.1. Given TR-STBC output processes $z_1(t)$ and $z_2(t)$ of (2.21) and (2.22), we now derive the Wiener filter tap weights for this architecture. To do this, we calculate the autocorrelation matrix $\boldsymbol{R}_\delta$ and cross-correlation vector $\boldsymbol{p}_\delta$ for outputs $z_1(t)$ and $z_2(t)$.

From (2.21) and (2.22), we see that both signal components $s_1(t)$ and $s_2(t)$ experience the same effective channel with polynomial representation

$$g(q, q^{-1})\ s_k(t) \stackrel{\triangle}{=} \left[h_1^*(q)\, h_1(q^{-1}) + h_2^*(q)\, h_2(q^{-1})\right]\ s_k(t), \qquad k = 1, 2\ . \tag{3.1}$$

Alternatively, we can write the effective channel response using convolutions in the time domain as

$$g(t) = \widetilde{h}_1^*(t) * h_1(t) + \widetilde{h}_2^*(t) * h_2(t) \tag{3.2}$$

where $h_1(t)$ and $h_2(t)$ are the two channel impulse responses. For appropriately ban-

dlimited channels, by the Sampling Theorem, we can represent the continuous-time system using an equivalent discrete-time system with no loss of information. If we oversample the continuous-time random processes $z_1(t)$ and $z_2(t)$ at P times the symbol rate, we can transform (2.21) and (2.22) into the equivalent discrete-time forms

$$z_1(k) = \sum_{\ell=0}^{v} g(k-\ell P)\, s_1(\ell) + \sum_{\ell=0}^{v_h P} h_1^*(\ell-k)\, n_1(\ell) + \sum_{\ell=0}^{v_h P} h_2(k-\ell)\, n_2^*(-\ell) \tag{3.3}$$

$$z_2(k) = \sum_{\ell=0}^{v} g(k-\ell P)\, s_2(\ell) + \sum_{\ell=0}^{v_h P} h_2^*(\ell-k)\, n_1(\ell) - \sum_{\ell=0}^{v_h P} h_1(k-\ell)\, n_2^*(-\ell) \tag{3.4}$$

where $h_1(k)$ and $h_2(k)$ both have delay spreads no greater than v_h symbols, and $v = \lceil (2v_hP-1)/P \rceil = 2v_h$.

Again, we use matrix and vector notation to simplify the analysis. If we stack N_fP oversampled observations of $z_1(k)$ and $z_2(k)$ into vectors $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$, respectively, we can write (3.3) and (3.4) in equivalent matrix-vector forms

$$\boldsymbol{z}_1 = \boldsymbol{G}\,\boldsymbol{s}_1 + \widetilde{\boldsymbol{H}}_1\,\boldsymbol{n}_1 + \boldsymbol{H}_2\,\widetilde{\boldsymbol{n}}_2 \tag{3.5}$$

$$\boldsymbol{z}_2 = \boldsymbol{G}\,\boldsymbol{s}_2 + \widetilde{\boldsymbol{H}}_2\,\boldsymbol{n}_1 - \boldsymbol{H}_1\,\widetilde{\boldsymbol{n}}_2\,. \tag{3.6}$$

Matrix $\boldsymbol{G}$ is of dimension $N_fP \times (N_f + v - 1)$ with

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{g}_1 & \boldsymbol{g}_2 & \cdots & \boldsymbol{g}_v & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{g}_1 & \boldsymbol{g}_2 & \cdots & \boldsymbol{g}_v & \boldsymbol{0} & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{g}_1 & \boldsymbol{g}_2 & \cdots & \boldsymbol{g}_v \end{bmatrix} \tag{3.7}$$

where length P vector

$$\boldsymbol{g}_k = [g[(k-1)P+P]\ g[(k-1)P+P-1]\ \cdots\ g[(k-1)P+1]]^T\ . \tag{3.8}$$

Matrix $\boldsymbol{H}_i$ is the $N_fP \times (N_fP + v_hP - 1)$ standard channel convolution matrix corre-

sponding to channel $h_i(t)$, $i = 1, 2$ with Toeplitz form

$$\boldsymbol{H}_i = \begin{bmatrix} h_i(1) & h_i(2) & \dots & h_i(v_h P) & 0 & \cdots & 0 \\ 0 & h_i(1) & h_i(2) & \cdots & h_i(v_h P) & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \vdots \\ 0 & \cdots & 0 & h_i(1) & h_i(2) & \cdots & h_i(v_h P) \end{bmatrix}. \tag{3.9}$$

Matrix $\widetilde{\boldsymbol{H}}_i$ is formed by replacing $h_i(t)$ by $\widetilde{h}_i^*(t)$ in (3.9).

With the TR-STBC outputs in matrix format, we now derive the Wiener solution for the TR-STBC SISO DFEs. To derive the Wiener solution for SISO DFE 1, we substitute $\boldsymbol{z}_1$ of (3.5) into the DFE autocorrelation matrix (2.58). Assuming the symbol and noise processes are mutually independent and i.i.d., we can show that

$$\boldsymbol{R}_{1,\delta} = \begin{bmatrix} \sigma_{s_1}^2 \boldsymbol{G}\boldsymbol{G}^H + \sigma_{n_1}^2 \widetilde{\boldsymbol{H}}_1 \widetilde{\boldsymbol{H}}_1^H + \sigma_{n_2}^2 \boldsymbol{H}_2 \boldsymbol{H}_2^H & -\sigma_{s_1}^2 \boldsymbol{G}(:, \delta+1 : \delta+N_b) \\ -\sigma_{s_1}^2 \boldsymbol{G}^H(:, \delta+1 : \delta+N_b) & \sigma_{s_1}^2 \boldsymbol{I}_{N_b} \end{bmatrix} \tag{3.10}$$

where $\sigma_{s_i}^2$ and $\sigma_{n_i}^2$, for $i = 1, 2$, represent the signal and noise powers. We use MATLAB notation $\boldsymbol{G}(:, \delta+1 : \delta+N_b)$ to denote columns $\delta+1$ through $\delta+N_b$ of matrix $\boldsymbol{G}$. Similarly, using the independence of the symbols and noise, the cross-correlation vector $\boldsymbol{p}_1$ has the form

$$\boldsymbol{p}_{1,\delta} = \begin{bmatrix} \sigma_{s_1}^2 \boldsymbol{G}(:, \delta) \\ \boldsymbol{0}_{N_b \times 1} \end{bmatrix} \tag{3.11}$$

where $\boldsymbol{0}_{N_b \times 1}$ is a length N_b zero column vector, and $\boldsymbol{G}(:, \delta)$ is column δ of matrix $\boldsymbol{G}$. Substituting (3.10) and (3.11) into (2.57), we can finally solve for $\boldsymbol{w}_{\text{opt},\delta}$. The optimal delay can be found by minimizing the MMSE equation with respect to δ

$$\delta_{1,\text{opt}} = \arg \min_{1 \le \delta \le N_f + v - 1 - N_b} \sigma_{s_1}^2 - \boldsymbol{p}_{1,\delta}^H \boldsymbol{R}_{1,\delta}^{-1} \boldsymbol{p}_{1,\delta} \tag{3.12}$$

leading to the optimal Weiner filter tap coefficients for the TR-STBC channel

$$\boldsymbol{w}_{1,\delta_{1,\text{opt}}} = \boldsymbol{R}_{1,\delta_{1,\text{opt}}}^{-1} \boldsymbol{p}_{1,\delta_{1,\text{opt}}}. \tag{3.13}$$

To derive the Wiener solution for SISO DFE 2, in a similar manner we substitute $\boldsymbol{z}_2$ of (3.6) into the DFE autocorrelation matrix (2.58). In this case, autocorrelation matrix $\boldsymbol{R}_{2,\delta}$ takes a very similar form to $\boldsymbol{R}_{1,\delta}$ of (3.10) with $\sigma_{s_1}^2$ replaced by $\sigma_{s_2}^2$ and new first element $\sigma_{s_2}^2 \boldsymbol{G}\boldsymbol{G}^H + \sigma_{n_1}^2 \widetilde{\boldsymbol{H}}_2 \widetilde{\boldsymbol{H}}_2^H + \sigma_{n_2}^2 \boldsymbol{H}_1 \boldsymbol{H}_1^H$. Likewise, crosscorrelation vector $\boldsymbol{p}_{2,\delta}$ has the same form as $\boldsymbol{p}_{1,\delta}$ of (3.11) with $\sigma_{s_1}^2$ replaced by $\sigma_{s_2}^2$.

It is important to note that we assume that $\sigma_{s_1}^2 = \sigma_{s_2}^2$ and $\sigma_{n_1}^2 = \sigma_{n_2}^2$. Also, we can show that

$$\widetilde{\boldsymbol{H}}_i \widetilde{\boldsymbol{H}}_i^H = \boldsymbol{H}_i \boldsymbol{H}_i^H, \qquad i = 1, 2 \; . \tag{3.14}$$

Therefore, after a few substitutions, it is clear that $\boldsymbol{R}_{1,\delta} = \boldsymbol{R}_{2,\delta}$ and $\boldsymbol{p}_{1,\delta} = \boldsymbol{p}_{2,\delta}$, and the Wiener solution for DFE 2 is identical to that of DFE 1.

To be able to compare the performance of this equalizer structure with the other ones, we add the MSEs of the two equalizers' outputs

$$J^{\text{sum}}(k) = \mathrm{E}\{|e_{1k}|^2\} + \mathrm{E}\{|e_{2k}|^2\} \, . \tag{3.15}$$

Using the Wiener filter in both equalizers, we finally find the sum MMSE is

$$J_{\min}^{\text{sum}} = \sigma_{s_1}^2 - \boldsymbol{p}_1^H \boldsymbol{R}_1^{-1} \boldsymbol{p}_1 + \sigma_{s_2}^2 - \boldsymbol{p}_2^H \boldsymbol{R}_2^{-1} \boldsymbol{p}_2 = 2 \left(\sigma_{s_1}^2 - \boldsymbol{p}_1^H \boldsymbol{R}_1^{-1} \boldsymbol{p}_1 \right) \tag{3.16}$$

where we have dropped the delays $\delta_{1,\text{opt}}$ and $\delta_{2,\text{opt}}$ from the notation.

If we independently adapt each of the SISO DFEs using the conventional LMS algorithm (2.32), the MSE dynamic convergence behavior of the two equalizers can be approximated using the Independence Method trajectory (2.66) or the Small Step-size Method trajectory (2.77) with $\boldsymbol{R}_{i,\delta}$ and $\boldsymbol{p}_{i,\delta}$ calculated using (3.10) and (3.11), respectively. If the two equalizers are initialized with different tap weights, the two trajectories will have to be computed separately and added, whereas if the two equalizers are initialized with the same tap weights, the trajectory for only one equalizer needs to be calculated and then doubled to yield the sum MSE trajectory. In this latter case,

using (3.16), we can write

$$J^{\text{sum}}(k) = J^{\text{sum}}_{\text{min}} + \mu_{\text{S}}\, J^{\text{sum}}_{\text{min}} \sum_{m=1}^{L} \frac{\lambda_m}{2 - \mu_{\text{S}}\, \lambda_m} + \sum_{m=1}^{L} \lambda_m \left[2\,|v_m(0)|^2 - \frac{\mu_{\text{S}}\, J^{\text{sum}}_{\text{min}}}{2 - \mu_{\text{S}}\, \lambda_m} \right] (1 - \mu_{\text{S}}\, \lambda_m)^{2k} \tag{3.17}$$

where λ_i, $i = 1, \ldots, L$ are the eigenvalues of $\boldsymbol{R}_1$ and μ_{S} is the common step-size for the two SISO DFEs.

3.1.2 Novel Hybrid DFE

As we have shown in Sec. 3.1.1, if we equalize the two output streams of the TR-STBC decoder using parallel SISO DFEs, the Wiener filter solutions for the two DFEs are identical. In this section we introduce a novel equalization scheme, shown in Fig. 3.3, which takes advantage of this property.

When studying a phase-splitting equalizer structure for a Carrierless Amplitude / Phase (CAP) modulated signal in Digital Subscriber Lines (DSL), Gao and Leung noticed that the Wiener filter solutions for the parallel in-phase and quadrature equalizers were identical, due to the identical second-order statistics of the incoming in-phase and quadrature random processes [38]. To take advantage of this property and the implicit coupling of the two incoming real processes, they proposed constraining the equalizer tap weights for the two parallel equalizers to be the same and to use the following modified LMS algorithm to update the shared tap weights

$$\overline{\boldsymbol{w}}(k+1) = \overline{\boldsymbol{w}}(k) + \mu\, [e_{Ik}\, \boldsymbol{u}_I(k) + e_{Qk}\, \boldsymbol{u}_Q(k)] \tag{3.18}$$

with errors

$$e_{Ik} = a_{k-\delta} - \overline{\boldsymbol{w}}^T(k)\, \boldsymbol{u}_I(k) \qquad e_{Qk} = b_{k-\delta} - \overline{\boldsymbol{w}}^T(k)\, \boldsymbol{u}_Q(k)\,. \tag{3.19}$$

Here a_k and b_k denote the real and imaginary parts of the transmitted symbols, and vectors $\boldsymbol{u}_I(k)$ and $\boldsymbol{u}_Q(k)$ contain the in-phase and quadrature observations in the tap delay lines.

Using analysis and simulations, they found that tap update equation (3.18) effectively couples the in-phase and quadrature equalizers together. Also, the averaging of the instantaneous gradient estimates for the two i.i.d. random processes reduces the gradient noise in the LMS algorithm, yielding a faster dynamic convergence rate than that achieved by the independent parallel equalizers. The resulting new hybrid equalizer with the modified LMS algorithm has less than half the misadjustment for the same initial convergence rate compared to the conventional LMS algorithm [38]. Alternatively, for the same misadjustment, the weights of the new adaptive hybrid equalizer initially converge faster than those of the independent equalizers.

Having observed similar coupled but identical signal statistics at the outputs of the TR-STBC-decoder, we now impose the same tap weight constraint on our parallel SISO DFEs and extend their modified LMS algorithm to accommodate complex random processes and a DFE structure. We call our new algorithm the *Combined Error-LMS (CE-LMS) algorithm*, with tap update equation

$$\overline{\boldsymbol{w}}(k+1) = \overline{\boldsymbol{w}}(k) + \mu\,[e_{1k}^{*}\,\boldsymbol{v}_1(k) + e_{2k}^{*}\,\boldsymbol{v}_2(k)] \tag{3.20}$$

and errors

$$e_{1k} = s_{1,\,k-\delta} - \overline{\boldsymbol{w}}^{H}(k)\,\boldsymbol{v}_1(k) \qquad e_{2k} = s_{2,\,k-\delta} - \overline{\boldsymbol{w}}^{H}(k)\,\boldsymbol{v}_2(k)\,. \tag{3.21}$$

To see the origin of this stochastic gradient algorithm, consider the sum MSE for a fixed common set of tap weights $\overline{\boldsymbol{w}}$

$$\begin{aligned} J^{\text{sum}}(\overline{\boldsymbol{w}}) &= \mathrm{E}\{|e_{1k}|^2 + |e_{2k}|^2\} = \mathrm{E}\{|s_1 - \overline{\boldsymbol{w}}^{H}\boldsymbol{v}_1|^2\} + \mathrm{E}\{|s_2 - \overline{\boldsymbol{w}}^{H}\boldsymbol{v}_2|^2\} \\ &= \sigma_{s_1}^2 + \sigma_{s_2}^2 - \overline{\boldsymbol{w}}^{H}\mathrm{E}\{\boldsymbol{v}_1 s_1^{*} + \boldsymbol{v}_2 s_2^{*}\} - \mathrm{E}\{s_1\boldsymbol{v}_1^{H} + s_2\boldsymbol{v}_2^{H}\}\overline{\boldsymbol{w}} \\ &\quad + \overline{\boldsymbol{w}}^{H}\mathrm{E}\{\boldsymbol{v}_1\boldsymbol{v}_1^{H} + \boldsymbol{v}_2\boldsymbol{v}_2^{H}\}\overline{\boldsymbol{w}} \end{aligned} \tag{3.22}$$

where we have temporarily dropped the delay δ from the notation. Again, taking the gradient of $J^{\text{sum}}(\overline{\boldsymbol{w}})$ with respect to $\overline{\boldsymbol{w}}^{*}$, we find

$$\nabla_{\overline{\boldsymbol{w}}^{*}} J^{\text{sum}}(\overline{\boldsymbol{w}}) = -\mathrm{E}\{\boldsymbol{v}_1 s_1^{*} + \boldsymbol{v}_2 s_2^{*}\} + \mathrm{E}\{\boldsymbol{v}_1\boldsymbol{v}_1^{H} + \boldsymbol{v}_2\boldsymbol{v}_2^{H}\}\,\overline{\boldsymbol{w}}. \tag{3.23}$$

Setting the gradient equal to the zero vector and solving for the tap weights, we find the optimal tap weights have the form

$$\overline{\boldsymbol{w}}_o = \left[\mathrm{E}\{\boldsymbol{v}_1\boldsymbol{v}_1^H + \boldsymbol{v}_2\boldsymbol{v}_2^H\}\right]^{-1}\mathrm{E}\{\boldsymbol{v}_1 s_1^* + \boldsymbol{v}_2 s_2^*\} \stackrel{\triangle}{=} \boldsymbol{R}_{\text{hybrid}}^{-1}\,\boldsymbol{p}_{\text{hybrid}} \tag{3.24}$$

where

$$\boldsymbol{R}_{\text{hybrid}} = \boldsymbol{R}_1 + \boldsymbol{R}_2 = 2\,\boldsymbol{R}_1 \qquad \boldsymbol{p}_{\text{hybrid}} = \boldsymbol{p}_1 + \boldsymbol{p}_2 = 2\,\boldsymbol{p}_1 \tag{3.25}$$

since $\boldsymbol{R}_1 = \boldsymbol{R}_2$ and $\boldsymbol{p}_1 = \boldsymbol{p}_2$. Finally, substituting $\overline{\boldsymbol{w}}_o$ back into MSE (3.22), the steady-state sum MMSE is

$$J_{\min}^{\text{sum}} = \sigma_{s_1}^2 + \sigma_{s_2}^2 - \boldsymbol{p}_{\text{hybrid}}^H\,\boldsymbol{R}_{\text{hybrid}}^{-1}\,\boldsymbol{p}_{\text{hybrid}} = 2\left(\sigma_{s_1}^2 - \boldsymbol{p}_1^H\,\boldsymbol{R}_1^{-1}\boldsymbol{p}_1\right) \tag{3.26}$$

which is the same as for the SISO DFEs.

If we adapt the hybrid DFE using the CE-LMS algorithm (3.20), the MSE dynamic convergence behavior of the equalizer can be approximated using the Independence Method trajectory (2.66) or the Small Step-size Method trajectory (2.77) with $\boldsymbol{R}$ and $\boldsymbol{p}$ replaced by $2\,\boldsymbol{R}_1$ and $2\,\boldsymbol{p}_1$, respectively. In this latter case, using (3.26), we have

$$\begin{aligned} J^{\text{sum}}(k) = J_{\min}^{\text{sum}} &+ \mu_{\text{H}}\,J_{\min}^{\text{sum}}\sum_{m=1}^{L}\frac{\lambda_m}{1-\mu_{\text{H}}\,\lambda_m} \\ &+ \sum_{m=1}^{L}\lambda_m\left[2\,|v_m(0)|^2 - \frac{\mu_{\text{H}}\,J_{\min}^{\text{sum}}}{1-\mu_{\text{H}}\,\lambda_m}\right](1-2\,\mu_{\text{H}}\,\lambda_m)^{2k} \end{aligned} \tag{3.27}$$

where $2\,\lambda_i,\ i = 1,\ldots,L$ are the eigenvalues of $2\,\boldsymbol{R}_1$ and μ_{H} is the step-size for the hybrid DFE.

3.1.3 MIMO DFE

As a last DFE structure, we consider the 2×2 MIMO DFE of Fig. 3.2. For the case of perfect knowledge of the channel, we demonstrate in this section that the output process vectors $\boldsymbol{v}_1(k)$ and $\boldsymbol{v}_2(k)$ are uncorrelated and orthogonal. This implies that two parallel SISO DFEs would perform at least as well as the MIMO DFE. However,

for the practical case of imperfect channel estimates at the receiver, the process vectors in general will be coupled. Therefore, joint processing of $\boldsymbol{v}_1(k)$ and $\boldsymbol{v}_2(k)$ could be beneficial and more robust to the potential loss of orthogonality, motivating us to study now the MIMO DFE structure.

If we concatenate the two SISO DFE vectors to form

$$\boldsymbol{v}_c(k) = \begin{bmatrix} \boldsymbol{v}_1(k) \\ \boldsymbol{v}_2(k) \end{bmatrix} \tag{3.28}$$

the MIMO version of the LMS tap update equation can be written

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + \mu\, \boldsymbol{v}_c(k)\, \boldsymbol{e}^H(k) \tag{3.29}$$

where error vector

$$\boldsymbol{e}(k) = \boldsymbol{s}(k-\delta) - \boldsymbol{W}^H(k)\, \boldsymbol{v}_c(k) \tag{3.30}$$

with $\boldsymbol{s}(k-\delta) = [s_{1,k-\delta}\ s_{2,k-\delta}]^T$. The corresponding MIMO sum MSE for a fixed set of tap weights $\boldsymbol{W}$ can be written

$$\begin{aligned} J^{\text{sum}}(\boldsymbol{W}) &= \mathrm{E}\{\|\boldsymbol{e}(k)\|^2\} = \mathrm{E}\{[\boldsymbol{s}(k-\delta) - \boldsymbol{W}^H\, \boldsymbol{v}_c(k)]^H [\boldsymbol{s}(k-\delta) - \boldsymbol{W}^H\, \boldsymbol{v}_c(k)]\} \\ &= \mathrm{E}\{\boldsymbol{s}^H(k-\delta)\, \boldsymbol{s}(k-\delta) - \boldsymbol{s}^H(k-\delta)\, \boldsymbol{W}^H\, \boldsymbol{v}_c(k) \\ &\quad - \boldsymbol{v}_c^H(k)\, \boldsymbol{W}\, \boldsymbol{s}(k-\delta) + \boldsymbol{v}_c^H(k)\, \boldsymbol{W}\, \boldsymbol{W}^H \boldsymbol{v}_c(k)\} \\ &= \mathrm{E}\{\boldsymbol{s}^H(k-\delta)\, \boldsymbol{s}(k-\delta)\} - \mathrm{Tr}(\boldsymbol{W}^H \mathrm{E}\{\boldsymbol{v}_c(k)\, \boldsymbol{s}^H(k-\delta)\}) \\ &\quad - \mathrm{Tr}(\mathrm{E}\{\boldsymbol{s}(k-\delta)\, \boldsymbol{v}_c^H(k)\}\, \boldsymbol{W}) + \mathrm{Tr}(\boldsymbol{W}^H \mathrm{E}\{\boldsymbol{v}_c(k)\, \boldsymbol{v}_c^H(k)\}\, \boldsymbol{W})\,. \end{aligned} \tag{3.31}$$

Taking the gradient of $\mathrm{MSE}_{\text{sum}}$ with respect to $\boldsymbol{W}^*$, we can show that

$$\nabla_{\boldsymbol{W}^*} J^{\text{sum}}(\boldsymbol{W}) = -\mathrm{E}\{\boldsymbol{v}_c(k)\, \boldsymbol{s}^H(k-\delta)\} + \mathrm{E}\{\boldsymbol{v}_c(k)\, \boldsymbol{v}_c^H(k)\}\, \boldsymbol{W}\,. \tag{3.32}$$

Setting the gradient equal to the zero matrix and solving for the tap weights, we find

the optimal tap weights have the form

$$\boldsymbol{W}_o = \left[\mathrm{E}\{\boldsymbol{v}_c(k)\,\boldsymbol{v}_c^H(k)\}\right]^{-1}\mathrm{E}\{\boldsymbol{v}_c(k)\,\boldsymbol{s}^H(k-\delta)\} \triangleq \boldsymbol{R}_{c,\delta}^{-1}\,\boldsymbol{P}_{c,\delta}\,. \tag{3.33}$$

If we let

$$\boldsymbol{R}_{c,\delta} = \mathrm{E}\{\boldsymbol{v}_c(k)\,\boldsymbol{v}_c^H(k)\} = \begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{R}_{12} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_2 \end{bmatrix} \tag{3.34}$$

by the definition of $\boldsymbol{v}_c(k)$, matrices $\boldsymbol{R}_1 = \boldsymbol{R}_2$ again have the form (3.10). Applying the independence of the signal and noise processes, the off-diagonal matrix $\boldsymbol{R}_{12}$ can be written as

$$\boldsymbol{R}_{12} = \mathrm{E}\{\boldsymbol{v}_1(k)\boldsymbol{v}_2^H(k)\} = \begin{bmatrix} \sigma_{n_1}^2\widetilde{\boldsymbol{H}}_1\widetilde{\boldsymbol{H}}_2^H - \sigma_{n_2}^2\boldsymbol{H}_2\boldsymbol{H}_1^H & \mathbf{O}_{N_fP\times N_b} \\ \mathbf{O}_{N_b\times N_fP} & \mathbf{O}_{N_b\times N_b} \end{bmatrix}. \tag{3.35}$$

But, because $\sigma_{n_1}^2 = \sigma_{n_2}^2$ and we can show that

$$\widetilde{\boldsymbol{H}}_1\widetilde{\boldsymbol{H}}_2^H = \boldsymbol{H}_2\boldsymbol{H}_1^H \tag{3.36}$$

the upper-left corner matrix reduces to a zero matrix, finally yielding

$$\boldsymbol{R}_{12} = \mathbf{O}_{(N_fP+N_b)\times(N_fP+N_b)}\,. \tag{3.37}$$

Similarly, the other off-diagonal matrix $\boldsymbol{R}_{21}$ is a zero matrix as well. These zero off-diagonal matrices demonstrate that the output process vectors $\boldsymbol{v}_1(k)$ and $\boldsymbol{v}_2(k)$ are uncorrelated.

Likewise, if we let

$$\boldsymbol{P}_{c,\delta} = \mathrm{E}\{\boldsymbol{v}_c(k)\,\boldsymbol{s}^H(k-\delta)\} = \begin{bmatrix} \boldsymbol{p}_1 & \boldsymbol{p}_{12} \\ \boldsymbol{p}_{21} & \boldsymbol{p}_2 \end{bmatrix} \tag{3.38}$$

by the definition of $\boldsymbol{v}_c(k)$, vectors $\boldsymbol{p}_1 = \boldsymbol{p}_2$ again have the form (3.11). Due to the independence of the signal and noise processes, we can show that the off-diagonal

vectors $\boldsymbol{p}_{12} = \boldsymbol{p}_{21} = \mathbf{0}_{(N_f P+N_b)\times 1}$.

Because of the block diagonal structure of $\boldsymbol{R}_{c,\delta}$ and $\boldsymbol{P}_{c,\delta}$, the Wiener filter tap weights (3.33) have the decoupled form

$$\boldsymbol{W}_o = \begin{bmatrix} \boldsymbol{R}_1^{-1}\,\boldsymbol{p}_{1,\delta} & \mathbf{0}_{(N_f P+N_b)\times 1} \\ \mathbf{0}_{(N_f P+N_b)\times 1} & \boldsymbol{R}_1^{-1}\,\boldsymbol{p}_{1,\delta} \end{bmatrix} \tag{3.39}$$

and the steady-state sum MMSE once again is

$$J_{\min}^{\text{sum}} = \sigma_{s_1}^2 + \sigma_{s_2}^2 - \text{Tr}(\boldsymbol{P}_{c,\delta}^H\,\boldsymbol{R}_{c,\delta}^{-1}\,\boldsymbol{P}_{c,\delta}) = 2\left(\sigma_{s_1}^2 - \boldsymbol{p}_1^H\boldsymbol{R}_1^{-1}\boldsymbol{p}_1\right)\,. \tag{3.40}$$

This implies that for the case of perfect channel knowledge there is no advantage in using a MIMO DFE over using simpler parallel SISO DFEs. Nevertheless, due to the potential loss of orthogonality for the estimated channel case, we study the MIMO DFE using analysis and simulations, as the joint processing of $\boldsymbol{v}_1(k)$ and $\boldsymbol{v}_2(k)$ could be beneficial.

If we adapt the MIMO DFE using the LMS algorithm (3.29), the MSE dynamic convergence behavior of the equalizer can be approximated using the Independence Method trajectory (2.65)–(2.66) with $\boldsymbol{K}(k) \stackrel{\triangle}{=} \text{E}\{[\boldsymbol{W}_o - \boldsymbol{W}(k)][\boldsymbol{W}_o - \boldsymbol{W}(k)]^H\}$, $\boldsymbol{R}$ replaced by $\boldsymbol{R}_c$, and $J_{\min}$ replaced by (3.40). Alternatively, we can compute the Small Step-size Method trajectory. For the MIMO case we have to modify (2.77) slightly, defining new eigen-decomposition $\boldsymbol{R}_c = \boldsymbol{Q}\,\boldsymbol{\Lambda}\,\boldsymbol{Q}^H$ and matrix $\boldsymbol{V}(0) \stackrel{\triangle}{=} \boldsymbol{Q}^H[\boldsymbol{W}_o - \boldsymbol{W}(0)]$. If we let $\boldsymbol{v}_m(0)$ be the m-th row of $\boldsymbol{V}(0)$, in (2.77) we must replace $|v_m(0)|^2$ by $\|\boldsymbol{v}_m(0)\|^2$. For the special case where we let only the top half of the first column and the bottom half of the second column of $\boldsymbol{W}(0)$ contain non-zero values, $\|\boldsymbol{v}_m(0)\|^2$ reduces back down to $|v_m(0)|^2$ defined previously. In this latter case, using (3.40), we can write

$$\begin{aligned} J^{\text{sum}}(k) = J_{\min}^{\text{sum}} &+ 2\,\mu_{\text{M}}\,J_{\min}^{\text{sum}} \sum_{m=1}^{L} \frac{\lambda_m}{2-\mu_{\text{M}}\,\lambda_m} \\ &+ \sum_{m=1}^{L} \lambda_m \left[2\,|v_m(0)|^2 - \frac{2\,\mu_{\text{M}}\,J_{\min}^{\text{sum}}}{2-\mu_{\text{M}}\,\lambda_m}\right](1-\mu_{\text{M}}\,\lambda_m)^{2k} \end{aligned} \tag{3.41}$$

where we have used the property that the eigenvalues of $\boldsymbol{R}_c$ are just the eigenvalues

of $\boldsymbol{R}_1$, λ_i, $i = 1, \ldots, L$, repeated twice, and we define μ_{M} to be the step-size for the MIMO DFE.

In this section we have introduced three alternative DFEs using architecture-specific versions of the LMS algorithm. Unfortunately, it is well known that the LMS algorithm tends to have a slow rate of convergence for channels with large autocorrelation eigenvalue spreads [27]. Therefore, before describing our analytical calculations and simulation results for these architectures, we first propose several methods to speed up the equalizer convergence rates in the next two sections.

3.2 TRAINING METHOD ENHANCEMENTS

Intelligent design of the training mode for adaptive equalization is very important for practical wireless communication systems. For mobile wireless systems, in particular, choosing the proper length and place of the training sequence in a data block is crucial, as the channel is constantly changing and the training sequence has to be repeatedly transmitted. Increasing the length of the training sequence increases the accuracy of the channel estimates and the equalizer performance but decreases the bandwidth efficiency of the system.

We now consider two training mode enhancements to improve the reliability of the channel estimates for the TR-STBC decoder and to simplify the equalizer post-processing as well as to increase the equalizer convergence rate while in training mode. The first enhancement centers on the TR-STBC packet structure, whereas the second involves jump-starting the training using the estimates of the channel impulse response and the Method of Steepest Descent.

3.2.1 Novel TR-STBC Data Packet Structure

Assuming a quasi-static but time-varying channel, the two major objectives of the training mode of the TR-STBC system are to estimate the channel impulse response (CIR) for the TR-STBC decoder and to tune the DFE taps to minimize the ISI before the decision device. As a brute force approach, we could design our data packet to have two training sequences, one allocated for each objective. For example, as shown in

Fig. 3.4 we could first send uncoded channel estimation sequences $\mathrm{T1_{CE}}$ and $\mathrm{T2_{CE}}$ that are L_{tce} symbols long and are stripped off the packet before the TR-STBC decoder and sent to a CIR estimation module. To train the DFE, separate training sequences $\mathrm{T1_{EQ}}$ and $\mathrm{T2_{EQ}}$, which are $L_{teq}/2$ symbols long, could be placed before the data payloads $D1$ and $D2$, respectively, and embedded into the TR-STBC block structure. After TR-STBC decoding, $\mathrm{T1_{EQ}}$ and $\mathrm{T2_{EQ}}$ would enter the DFE before the data payloads. Only three zero guard intervals of length $L_g \geq v_h - 1$ would be required for this packet structure.

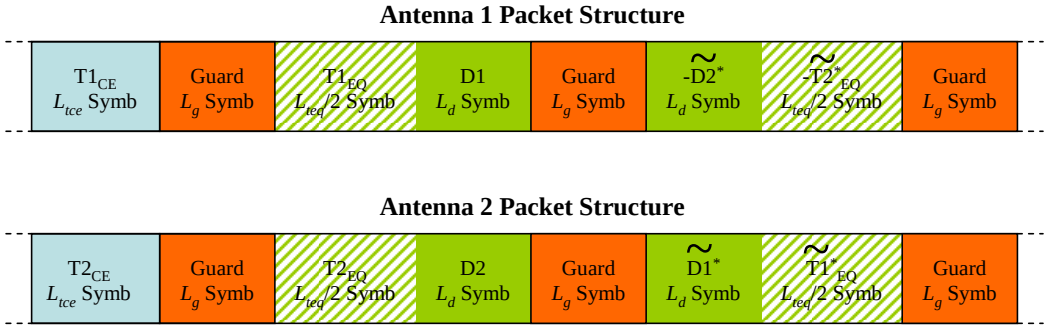

Figure 3.4 Brute force TR-STBC data packet with separate training sequences

To choose good channel estimation sequences $\mathrm{T1_{CE}}$ and $\mathrm{T2_{CE}}$, as discussed in Appendix A, a widely used technique is to perform an exhaustive search over all $\mathrm{ord}(s)^{2L_{tce}}$ possible pairs of symbol sequences $\{\mathrm{T1_{CE}}, \mathrm{T2_{CE}}\}$, where $\mathrm{ord}(s)$ is the constellation size of s. The sequence pair which minimizes the MSE (A.8) between the true channel and the block least-squares estimate of the channel is selected. For example, for $L_{tce} = 20$, over 10^{23} possible sequence pairs must be checked. Due to this prohibitively large search and the advantages of the TR-STBC structure, we instead introduce the suboptimal approach proposed by Fragouli *et al.* in [39].

As mentioned in Appendix A, Fragouli *et al.* impose the TR-STBC block structure on the channel estimation sequences. After solving for the required constraints, they show that a single training subsequence that is only $L_{tce}/2$ symbols long can be repeated in two TR-STBC encoded blocks. The resulting channel estimation sequences transmitted from the two antennas have the forms $[\mathrm{T1_{CE}} \ -\mathrm{T1_{CE}^*}]$ and $[\widetilde{\mathrm{T1}}_{\mathrm{CE}} \ \widetilde{\mathrm{T1}}_{\mathrm{CE}}^*]$. For $L_{tce} = 20$ and QPSK, only around 10^6 possible subsequences must be tested to find

the optimal $T1_{CE}$. The cost of adding this TR-STBC constraint to the sequences is a slightly higher MMSE value. If this new TR-STBC structure is added to the channel estimation sequences, the data packet assumes the form shown in Fig. 3.5. Note that an additional guard interval is required between the two channel estimation subsequences at the beginning of the data packet.

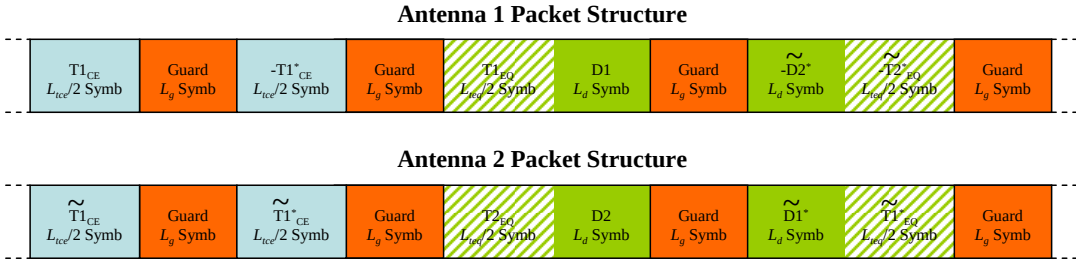

Figure 3.5 Brute force TR-STBC data packet with separate TR-STBC encoded channel estimation sequence

Our novel data packet structure comes from the observation that the TR-STBC encoded channel estimation sequences and the TR-STBC encoded equalizer training sequences can be combined into a single pair of sequences which can perform both training mode functions. If we adopt the packet structure show in Fig. 3.6, the observations corresponding to the training sequences at the ends of packets can be extracted prior to the TR-STBC decoder to form least-squares estimates of the two CIRs. These same observations can be sent through the TR-STBC decoder as part of the data packet and reused as the training sequences for the DFEs. At the TR-STBC decoder output, the training sequence appears first followed by the data payload.

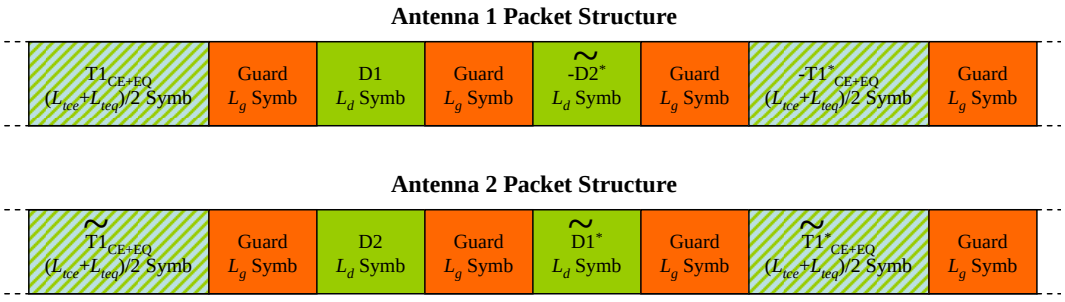

Figure 3.6 Our novel TR-STBC data packet structure with combined training sequences

Our novel packet structure has many advantages. First, the aggregation of the channel estimation sequences and equalizer training sequences into dual-purpose sequences potentially reduces the overall number of training symbols required per data packet, increasing the bandwidth efficiency of the system. Second, in the transmitter all of the data can be passed through a single TR-STBC encoder without having to append uncoded symbol sequences, reducing the hardware complexity. Third, the training sequences are in the correct positions for equalizer training, so that in the receiver the TR-STBC decoder outputs can be sent directly to the DFEs without extra post-processing (unlike for the data packet structures shown in Figs. 3.4 and 3.5).

Finally, although we have assumed that the channel is quasi-static or constant over the data packet, for practical mobile wireless channels the CIR will vary slowly across the packet. Having the channel estimation subsequences separated at the ends of the packets has an averaging effect on the channel estimates. Because observations at the extremities of the packet are used to form the estimates, the channel estimates are better able to take into account the time-variations of the channel. This robustness is not intrinsic in the conventional packet structures of Figs. 3.4 and 3.5. We now outline our second training mode enhancement, which is specific to the DFE training operation.

3.2.2 Kick-Start Method of Steepest Descent

As just mentioned, the first step in processing a TR-STBC data packet is to use the training symbols to form CIR estimates $\widehat{h}_1(k)$ and $\widehat{h}_2(k)$. Theoretically, given these estimates, we could use them in place of the true CIRs $h_1(k)$ and $h_2(k)$ to form estimated matrices $\widehat{\boldsymbol{G}}$ of (3.7) and $\widehat{\boldsymbol{H}}_i$ of (3.9), for $i = 1, 2$. These estimated matrices could in turn be used to form estimated autocorrelation matrix $\widehat{\boldsymbol{R}}_1$ of (3.10) and crosscorrelation vector $\widehat{\boldsymbol{p}}_1$ of (3.11). Finally, after inverting $\widehat{\boldsymbol{R}}_1$ the Wiener filter tap weights could be estimated using $\widehat{\boldsymbol{w}}_{1,\delta_{1,\text{opt}}}$ of (3.13), eliminating the need for a training sequence to tune the DFE tap weights.

In practice, there are two reasons why this method does not work. First, given a finite length channel estimation sequence in a time-varying channel, the estimates

$\widehat{h}_1(k)$ and $\widehat{h}_2(k)$ will include residual errors, leading to errors in the estimated Wiener filter taps weights $\widehat{\boldsymbol{w}}_{1,\delta_{1,\text{opt}}}$. Thus, further adaptive tuning of the DFE taps, using an equalizer training sequence, is required to minimize the ISI at the detector input. Second, in a practical TR-STBC system, the computations must be processed in real time. Unfortunately, the inversion of $\widehat{\boldsymbol{R}}_1$ requires approximately $(N_f P + N_b)^3$ complex multiplies, which is very difficult to calculate in real time. In a practical FPGA-based MIMO testbed, we have found that this number of required real-time multiplies is prohibitively large. Because we are interested in designing a system with reasonable computational complexity, we propose the following kick-start method using the CIR estimates and the Method of Steepest Descent.

We first use the CIR estimates $\widehat{h}_1(k)$ and $\widehat{h}_2(k)$ to form estimated autocorrelation matrix $\widehat{\boldsymbol{R}}_1$ and crosscorrelation vector $\widehat{\boldsymbol{p}}_1$. Then, because our goal is to estimate $\widehat{\boldsymbol{w}}_{1,\delta_{1,\text{opt}}}$, instead of directly inverting $\widehat{\boldsymbol{R}}_1$ in a batch calculation, we apply the Method of Steepest Descent recursion (2.30)

$$\widehat{\boldsymbol{w}}(k+1) = \mu\,\widehat{\boldsymbol{p}}_1 + [\boldsymbol{I}_{N_f P+N_b} - \mu\,\widehat{\boldsymbol{R}}_1]\,\widehat{\boldsymbol{w}}(k)\,. \tag{3.42}$$

We iterate this equation a few times to converge quickly towards the Wiener filter taps using our block-based estimates of the true gradient. Using Monte Carlo simulations in Sec. 3.5, we show that the Method of SD requires only two or three iterations to "kick-start" the MSE trajectory quickly towards the MMSE. However, because the Method of SD recursion involves matrix multiplication, to reduce the complexity to facilitate real-time implementation, after only a few iterations we switch to the conventional LMS training update equation to process the DFE training sequence.

Regarding initializing the tap weights $\widehat{\boldsymbol{w}}(0)$, we suggest commencing with all zero tap weights. Assuming that the time variations of the channel are reasonably slow from data packet to data packet, after the first packet has been processed, $\widehat{\boldsymbol{w}}(0)$ should be initialized with the final tap weights corresponding to the previous packet. Also, care has to be taken regarding selecting the optimal delay δ to form $\widehat{\boldsymbol{p}}_1$.

This new Method of SD kick-start method has the advantage of accelerating the DFE training convergence, reducing the required length of the training sequence. Also,

because the Method of SD update algorithm does not depend on the TR-STBC outputs, it can be run concurrently with the TR-STBC decoding, after the channel estimates have been calculated. Finally, as we have shown in Sec. 3.1.1 that the Wiener filter tap weights are the same for both SISO DFEs, only one set of kick-start calculations are required for both DFEs.

3.3 EXTENSION OF HYBRID ALGORITHMS

Due to the generally slow convergence behavior of the LMS algorithm, we introduced the alternative NLMS and RLS algorithms in Secs. 2.3.4 and 2.3.5, respectively. As with the LMS algorithm, we can again implement these algorithms using the three DFE architectures proposed in Sec. 3.1. The tap update equations used for the parallel SISO DFEs and the MIMO DFE are straight-forward extensions of the traditional NLMS and RLS update equations reviewed in Secs. 2.3.4 and 2.3.5. In this section we concentrate on deriving novel Combined Error forms of these algorithms tailored to the hybrid DFE structure, again taking advantage of the common second-order statistics of the two TR-STBC output streams.

3.3.1 Combined Error-NLMS Algorithm

For two independently-operating parallel SISO DFEs adapted using the NLMS algorithm, as shown in Sec. 2.3.4 the distinct equalizer tap weights are updated using:

$$\boldsymbol{w}_i(k+1) = \boldsymbol{w}_i(k) + \frac{\mu_o}{\|\boldsymbol{v}_i(k)\|^2 + \delta}\, e_{ik}^*\, \boldsymbol{v}_i(k), \qquad \text{for } i = 1, 2. \tag{3.43}$$

However, because the two TR-STBC output streams share the same second-order statistics and Wiener filter tap weights, we apply the Combined Error design technique advocated in Sec. 3.1.2 to the NLMS algorithm. We again constrain the equalizer tap weights for the two parallel DFEs to be the same and now use the following Combined Error-NLMS (CE-NLMS) algorithm to update the shared tap weights

$$\overline{\boldsymbol{w}}(k+1) = \overline{\boldsymbol{w}}(k) + \frac{\mu_o}{\|\boldsymbol{v}_1(k)\|^2 + \delta}\, e_{1k}^*\, \boldsymbol{v}_1(k) + \frac{\mu_o}{\|\boldsymbol{v}_2(k)\|^2 + \delta}\, e_{2k}^*\, \boldsymbol{v}_2(k) \tag{3.44}$$

Table 3.1 Summary of Conventional RLS Algorithm for Parallel SISO DFEs

Conventional RLS Algorithm (for $i = 1, 2$)
Initial conditions: $\boldsymbol{w}_i(0) = \mathbf{0}$ $\boldsymbol{P}_i(0) = \delta^{-1}\boldsymbol{I}$ where δ is a small positive constant for high SNR
To update taps at each iteration, compute $\boldsymbol{\pi}_i(k) = \boldsymbol{P}_i(k-1)\,\boldsymbol{v}_i(k)$ $\boldsymbol{k}_i(k) = \frac{\boldsymbol{\pi}_i(k)}{\lambda + \boldsymbol{v}_i^H(k)\,\boldsymbol{\pi}_i(k)}$ $\epsilon_{ik} = s_{i,k-\delta} - \boldsymbol{w}_i^H(k-1)\,\boldsymbol{v}_i(k)$ $\boldsymbol{w}_i(k) = \boldsymbol{w}_i(k-1) + \boldsymbol{k}_i(k)\,\epsilon_{ik}^*$ $\boldsymbol{P}_i(k) = \lambda^{-1}\boldsymbol{P}_i(k-1) - \lambda^{-1}\boldsymbol{k}_i(k)\,\boldsymbol{v}_i^H(k)\,\boldsymbol{P}_i(k-1)$

with errors e_{1k} and e_{2k} defined in (3.21).

Once again, averaging the instantaneous gradient estimates for the two i.i.d. random processes reduces the gradient noise in the NLMS algorithm, resulting in a faster dynamic convergence rate than that achieved by the independent parallel equalizers. In this thesis we only use Monte Carlo simulations to evaluate the convergence behavior of the various forms of the NLMS algorithm. But, as mentioned in Sec. 2.4, the analytical techniques proposed in [30] could be used to calculate theoretical trajectories similar to those for the LMS algorithm.

3.3.2 Averaged Estimate-RLS Algorithm

As a last adaptive filtering algorithm option, we consider the RLS algorithm. As shown in Sec. 2.3.5 for two independently-operating parallel SISO DFEs the distinct equalizer tap weights in this case are updated using the recursions summarized in Table 3.1. To form a Combined Error version of the RLS algorithm, note that for the CE-LMS and CE-NLMS algorithms, tap weight vector $\overline{\boldsymbol{w}}(k)$ is the sole algorithm parameter other than the observations or previous decisions that is passed from one iteration to the next. From Table 3.1 we see that two parameters, tap weight vector $\boldsymbol{w}(k)$ and inverse correlation matrix $\boldsymbol{P}(k)$, are passed from one iteration to the next.

Using our Combined Error approach to the RLS algorithm running on the hybrid DFE structure, we would like to take into account the common second-order statistics of the TR-STBC outputs. As a first Combined Error modification, we constrain pa-

Table 3.2 Summary of AE-RLS Algorithm for Hybrid DFE

Averaged Estimate-RLS Algorithm	
Initial conditions:	
$\boldsymbol{w}_{\mathrm{AE}}(0) = \mathbf{0}$	
$\boldsymbol{P}_{\mathrm{AE}}(0) = \delta^{-1}\boldsymbol{I}$	
where δ is a small positive constant for high SNR	
To update taps at each iteration, compute	
$\boldsymbol{\pi}_1(k) = \boldsymbol{P}_{\mathrm{AE}}(k-1)\,\boldsymbol{v}_1(k)$	$\boldsymbol{\pi}_2(k) = \boldsymbol{P}_{\mathrm{AE}}(k-1)\,\boldsymbol{v}_2(k)$
$\boldsymbol{k}_1(k) = \frac{\boldsymbol{\pi}_1(k)}{\lambda + \boldsymbol{v}_1^H(k)\,\boldsymbol{\pi}_1(k)}$	$\boldsymbol{k}_2(k) = \frac{\boldsymbol{\pi}_2(k)}{\lambda + \boldsymbol{v}_2^H(k)\,\boldsymbol{\pi}_2(k)}$
$\epsilon_{1k} = s_{1,k-\delta} - \boldsymbol{w}_{\mathrm{AE}}^H(k-1)\,\boldsymbol{v}_1(k)$	$\epsilon_{2k} = s_{2,k-\delta} - \boldsymbol{w}_{\mathrm{AE}}^H(k-1)\,\boldsymbol{v}_2(k)$
$\boldsymbol{w}_{\mathrm{AE}}(k) = \boldsymbol{w}_{\mathrm{AE}}(k-1) + \frac{1}{2}\left[\boldsymbol{k}_1(k)\,\epsilon_{1k}^* + \boldsymbol{k}_2(k)\,\epsilon_{2k}^*\right]$	
$\boldsymbol{P}_{\mathrm{AE}}(k) = \lambda^{-1}\boldsymbol{P}_{\mathrm{AE}}(k-1) - \frac{1}{2\lambda}\left[\boldsymbol{k}_1(k)\,\boldsymbol{v}_1^H(k) + \boldsymbol{k}_2(k)\,\boldsymbol{v}_2^H(k)\right]\boldsymbol{P}_{\mathrm{AE}}(k-1)$	

rameters $\boldsymbol{w}_1(k) = \boldsymbol{w}_2(k) \stackrel{\triangle}{=} \boldsymbol{w}_{\mathrm{AE}}(k)$ and $\boldsymbol{P}_1(k) = \boldsymbol{P}_2(k) \stackrel{\triangle}{=} \boldsymbol{P}_{\mathrm{AE}}(k)$. Just as we have averaged the update increments $e_{ik}^*\,\boldsymbol{v}_i(k)$, for $i = 1, 2$, together in the CE-LMS algorithm to form $\overline{\boldsymbol{w}}(k)$, we average increments $\boldsymbol{k}_i(k)\,\epsilon_{ik}^*$ and $\boldsymbol{k}_i(k)\,\boldsymbol{v}_i^H(k)$ to form $\boldsymbol{w}_{\mathrm{AE}}(k)$ and $\boldsymbol{P}_{\mathrm{AE}}(k)$, respectively. The update equations for the new hybrid version, which we call the *Averaged Estimate-RLS (AE-RLS) algorithm*, are summarized in Table 3.2. This new AE-RLS algorithm requires approximately the same computational complexity as two RLS algorithms running in parallel, but, as we show using Monte Carlo simulations in Sec. 3.5, the parameter estimate averaging and tap weight equality constraint lead to increased convergence speeds for our novel algorithm.

3.3.3 Combined Estimate-RLS Algorithm

As a second Combined Error modification of the RLS algorithm, instead of introducing a heuristic averaging of certain parameter values as in the AE-RLS algorithm, we derive a new RLS algorithm from first principles, while enforcing the constraint that the equalizer taps for the two parallel SISO DFEs must be the same. In this case, the tap input vectors $\boldsymbol{v}_1(k)$ and $\boldsymbol{v}_2(k)$ are used jointly to minimize a weighted least squares criterion. In Appendix B we give a detailed derivation of this new *Combined Estimate-RLS (CE-RLS) algorithm* which is summarized in Table 3.3. Comparing the AE-RLS algorithm with the CE-RLS algorithm, we see that the latter algorithm has slightly greater computationally complexity. In particular, the inversion of a 2×2 matrix is required to form $\boldsymbol{K}(k)$. We will compare the complexities in greater detail in the next

Table 3.3 Summary of CE-RLS Algorithm for Hybrid DFE

Combined Estimate-RLS Algorithm
Initial conditions: $\boldsymbol{w}_{\mathrm{CE}}(0) = \mathbf{0}$ $\boldsymbol{P}_{\mathrm{CE}}(0) = \delta^{-1}\boldsymbol{I}$ where δ is a small positive constant for high SNR
To update taps at each iteration, compute $\boldsymbol{\Pi}(k) = \boldsymbol{P}_{\mathrm{CE}}(k-1)\,\boldsymbol{V}(k)$ where $\boldsymbol{V}(k) = [\boldsymbol{v}_1(k)\ \boldsymbol{v}_2(k)]$ $\boldsymbol{K}(k) = \boldsymbol{\Pi}(k)\,[\lambda\boldsymbol{I}_2 + \boldsymbol{V}^H(k)\,\boldsymbol{\Pi}(k)]^{-1}$ $\boldsymbol{\epsilon}(k) = \boldsymbol{s}(k) - \boldsymbol{V}^T(k)\,\boldsymbol{w}^*_{\mathrm{CE}}(k-1)$ where $\boldsymbol{s}(k) = [s_{1,k-\delta}\ s_{2,k-\delta}]^T$ $\boldsymbol{w}_{\mathrm{CE}}(k) = \boldsymbol{w}_{\mathrm{CE}}(k-1) + \boldsymbol{K}(k)\,\boldsymbol{\epsilon}^*(k)$ $\boldsymbol{P}_{\mathrm{CE}}(k) = \lambda^{-1}\boldsymbol{P}_{\mathrm{CE}}(k-1) - \lambda^{-1}\boldsymbol{K}(k)\,\boldsymbol{V}^H(k)\,\boldsymbol{P}_{\mathrm{CE}}(k-1)$

section.

More importantly, due to the clever coupling of information between the two equalizers in the hybrid DFE, as we show using Monte Carlo simulations in Sec. 3.5, the CE-RLS algorithm leads to an even faster convergence rate than the AE-RLS algorithm. In this thesis we only use Monte Carlo simulations to evaluate the convergence behavior of the various forms of the RLS algorithm. But, the analytical techniques proposed in [27, Sec. 9.7] could be used to calculate theoretical trajectories as we have done for the different variations of the LMS algorithm.

3.4 EQUALIZER COMPLEXITY ANALYSIS

With the performance and bandwidth gains of MIMO systems unfortunately comes increased complexity. To build real-time MIMO hardware, care has to be taken to engineer in the required computational power. Fortunately, most of the increased complexity of our MIMO equalization systems over conventional SISO systems involves matrix and vector multiplication, which can be easily parallelized.

In this section we measure the complexity of our equalizer algorithms and architectures in terms of Multiply-ACcumulate (MAC) operations. For example, previous researchers have estimated the processing requirement for a 2-channel Alamouti hardware prototype of a narrowband communication system to be around 3 billion MACs per second, whilst an implemented TR-STBC system requires close to 12 times this [21].

Here, we discuss the practical feasibility of implementing our algorithms and archi-

tectures on the existing STAR TR-STBC platform of [11]. In this case, we consider using a programmable Field Programmable Gate Array (FPGA) involving reconfigurable logic to construct the TR-STBC decoder, allowing greater coding complexity than traditional DSP-based platforms. In particular, we evaluate an Altera EP1S25 FPGA with in-built DSP blocks and multipliers. This FPGA operating at 120MHz is theoretically able to deliver (in terms of MACs):

- 10 DSP blocks: 4800MMACs
- 80 9-bit multipliers: 4800MMACs
- 25660 uncommitted logic elements: 7900MMACs

which totals to well over 17 billion MACs per second. (Although in an actual implementation usually half of the hardware elements would be reserved for optimization purposes.)

Note that the EP1S25 was state-of-the-art four years ago. But, even with all of its processing power, based upon our experimental work with the STAR platform, we have found that to implement all of the STBC decoding as well the equalizer inside the FPGA, it would be necessary to upgrade to a current state-of-the-art FPGA which would be multiple times larger and faster. To overcome the limitation of the existing FPGA, in our hypothetical study all of the MIMO processing is handled by the EP1S25 FPGA apart from the adaptive filtering which is implemented using an auxiliary TMS320C6714 DSP, which is capable of 2880 MMACS in the form of 4 x 16-bit MACs per 720MHz clock cycle.

The first step in TR-STBC decoding is to estimate the channel based on the observations due to the channel estimation training sequence. Using the least-squares approach of Appendix A, as shown in (A.7) the estimated CIR $\widehat{\boldsymbol{H}}$ can be formed by multiplying the $(L_{tce} - v_h + 1) \times P$ observation matrix $\boldsymbol{Y}$ by the $2v_h \times (L_{tce} - v_h + 1)$ pseudo-inverse training matrix $(\boldsymbol{S}^H \boldsymbol{S})^{-1} \boldsymbol{S}^H$. Fortunately, this fixed pseudo-inverse matrix can be pre-computed and stored in read-only memory (ROM). To create the decoupled decoder output streams using the estimates of the CIR, as shown in (2.21)–(2.22) the next step is to filter the observation blocks and their time-reversed versions

using the estimated CIRs $\widehat{h}_1(t)$ and $\widehat{h}_2(t)$.

Because this thesis is concerned with practical equalization strategies for TR-STBC decoded data, we concentrate on the equalizer part of the receiver. We now look at the multiplies and adds required by our new equalization algorithms, keeping implementation on the STAR platform in mind. We first evaluate the complexity of the training method enhancements of Sec. 3.2 followed by the equalization algorithms of Secs. 3.1 and 3.3.

3.4.1 Training Method Enhancements Complexity Analysis

A comparison of the computational complexity resulting from using the "Brute Force" packet structure of Fig. 3.4 versus the novel Prototype structure of Fig. 3.6 requires knowledge of the particular packet parameter values and equalization algorithms to be used. In this section and the next, we quantify the arithmetic operations that are required to make the various component calculations. Aggregating these component calculations, a comparison of packet structures can be made.

The first component is the possible Kick-Start Method of SD which requires additional computations over conventional equalizer training. To perform the Method of SD algorithm, we first need to construct the $\boldsymbol{R}_\delta$ matrix of (3.10) for the SD tap-update equation (2.30). Matrix $\boldsymbol{R}_\delta$ comprises a combination of matrices $\boldsymbol{G}$, $\boldsymbol{H}_i$ and their products. Matrix $\boldsymbol{G}$ is formed using the TR-STBC channel $g(t)$ of (3.8), which according to (3.1) is the sum of the convolutions of $h_1(t)$ and $h_2(t)$ with their time reversed and conjugated counterparts, all of length v_hP. Taking into account the conjugate symmetry of the convolution, the total number of complex multiplies and adds needed for each convolution are $(v_hP)^2/2$ and $v_hP\,(v_hP-1)/2$, respectively. Thus, to form the discrete samples of $g(t)$ requires $(v_hP)^2$ complex multiples and $(v_hP)^2$ complex adds. Matrices $\boldsymbol{G}$ and $\boldsymbol{H}$ of (3.7) and (3.9) are respectively of sizes $N_fP \times (N_f+v-1)$ and $N_fP \times (N_fP+v_hP-1)$. Because matrix $\boldsymbol{G}$ is block Toeplitz, the matrix product $\boldsymbol{G}\boldsymbol{G}^H$ requires approximately $(v^2P^2+vP)/2$ complex multiplies and $[(v-1)^2P^2+(v-1)P]/2$ complex adds to compute the distinct nonzero matrix entries.[1] To reduce multiplies, we pre-multiply $\boldsymbol{G}$ by σ_{s_1} before forming the product matrix $\boldsymbol{G}\boldsymbol{G}^H$, which requires

[1] Note that the number of computations to form $\boldsymbol{G}\boldsymbol{G}^H$ can be further reduced slightly for $N_f < v$.

$vP/2$ complex multiplies. Because $\boldsymbol{H}_i\boldsymbol{H}_i^H$ is Hermitian Toeplitz, we can show that its first row can be constructed using the last v_hP entries of $h_i(t) * \widetilde{h}_i^*(t)$ followed by $(N_f - v_h)/P$ zeros. Because the convolution $h_i(t) * \widetilde{h}_i^*(t)$ was already computed to form $g(t)$, no computations are required to form $\boldsymbol{H}_i\boldsymbol{H}_i^H$. To accommodate the $\sigma_{n_i}^2$ factor, we multiply the first v_hP entries of the first row of $\boldsymbol{H}_i\boldsymbol{H}_i^H$ by $\sigma_{n_i}^2$, requiring $v_hP/2$ complex multiplies. Using the definition of $\boldsymbol{R}_\delta$ (3.10), which includes multiplying $\boldsymbol{G}$ and $\boldsymbol{H}$ by the scalar real-valued signal and noise powers, the total number of multiplies and adds required to construct $\boldsymbol{R}_\delta$ are shown in Table 3.4. To simplify the summation we use $v = 2v_h$.

Table 3.4 Computational Complexity of $\boldsymbol{R}_\delta$

Operation	**Complex Multiplies**	**Complex Adds**
$g(\cdot)$ formation	$(v_hP)^2$	$(v_hP)^2$
$\sigma_{s_1}^2\boldsymbol{G}\boldsymbol{G}^H$ $(*)$	$v^2P^2/2 + vP$	$[(v-1)^2P^2 + (v-1)P]/2$
$\sigma_{n_1}^2\widetilde{\boldsymbol{H}}_1\widetilde{\boldsymbol{H}}_1^H$	$v_hP/2$	–
$\sigma_{n_2}^2\boldsymbol{H}_2\boldsymbol{H}_2^H$	$v_hP/2$	–
$\sigma_{s_1}^2\boldsymbol{G}(:, \delta+1 : \delta+N_b)$	done in $(*)$	–
$\sigma_{s_1}^2\boldsymbol{G}^H(:, \delta+1 : \delta+N_b)$	done in $(*)$	–
$\sigma_{s_1}^2 I_{N_b}$	–	–
Total	$3\,(v_h^2P^2 + v_hP)$	$[(6v_h^2 - 4v_h + 1)P^2$ $+(2v_h - 1)P]/2$

For example, for our STAR channel simulation with $v_h = 4$, $v = 8$, $P = 5$ and $N_f = N_b = 5$, we find that 400 multiplies are needed to form matrix $\boldsymbol{G}$. The parameter values yield a size 25×12 $\boldsymbol{G}$ matrix and a size 25×44 $\boldsymbol{H}$ matrix. Multiplying the $\boldsymbol{G}$ matrix with its complex conjugate transpose requires 820 complex multiplies and 630 complex adds. By substituting the STAR channel simulation parameters values into Table 3.4, we conclude that a total of 1260 complex multiplies and 1030 adds are needed to form $\boldsymbol{R}_\delta$.

After forming $\boldsymbol{R}_\delta$, we cycle through the SD tap-update algorithm (3.42) a few times to move down the steepest part of its convergence curve. To simplify the notation, we let $L = N_fP + N_b$. Using the $L \times L$ matrix $\boldsymbol{R}_\delta$ and the length L vectors $\boldsymbol{p}$ and $\boldsymbol{w}$, we first must compute the constant Hermitian matrix $[\boldsymbol{I}_L - \mu\,\boldsymbol{R}_\delta]$ and vector $\mu\,\boldsymbol{p}_\delta$, requiring a total of $L^2/4 + L/2$ complex multiplies and $L/2$ complex adds. Then, to update the tap weights during each iteration, we require a total of L^2 complex multiplies and L^2 complex adds. This computational complexity is about an order of magnitude greater

Table 3.5 Complexity Comparison of SISO, MISO and MIMO DFE Output Calculations

System	Complex Multiplies	Complex Adds
SISO	$N_f P + N_b$	$N_f P + N_b - 1$
MISO $m \times 1$	$m\,N_f P + N_b$	$m\,N_f P + N_b - 1$
MIMO $m \times n$	$mn\,N_f P + n^2 N_b$	$mn\,N_f P + n^2 N_b - n$

than that of the LMS tap-update algorithm in Table 2.3.

3.4.2 Equalizer Complexity Comparison

There are two complexity components for our adaptive equalizers: the complexity of computing the equalizer output for a given equalizer architecture and the complexity of recursively updating the equalizer tap weights using a particular adaptive filtering algorithm. In this section we consider both components for the three DFE structures introduced in Sec. 3.1, running the various adaptive algorithms we have proposed. For a given set of tap weights, Table 3.5 shows the number of complex multiplies and adds required to compute the equalizer output for generic fractionally-spaced SISO, MISO and MIMO DFEs.[2] For the 2×1 TR-STBC case, the number of calculations required to form the equalizer output of our three DFE structures are summarized in Table 3.6. We see that the parallel SISO DFEs and the hybrid DFE have the same complexity which is significantly less than that of the MIMO DFE.

Regarding the adaptive algorithms, the computational complexities of the LMS, NLMS, and RLS algorithms for a single SISO DFE are summarized in Tables 2.3, 2.4 and 2.6, where $L = N_f P + N_b$. Most of the algorithms for the hybrid DFE have similar complexities to those for the two parallel SISO DFEs. This is logical since the hybrid algorithms operate the two SISO DFEs concurrently. One special case is the Combined

[2]Note that after the slicer, the symbols that are sent into the feedback portion of the DFE can be represented using integers. Therefore, to compute the feedback filter output, only floating point (tap weights) times integer (feedback samples) multiplies are required, unlike the full floating point multiplies that are required in the feedforward filter. For the cases of BPSK and QPSK data, the multiplies in the feedback filter can be completely replaced by floating point additions, as the symbols are composed only of ± 1 and $\pm j$. In this thesis, we do not distinguish between floating point-integer multiplies and full floating point multiplies.

Table 3.6 Complexity Comparison of DFE Output Calculations for 2×1 TR-STBC System

System	Complex Multiplies	Complex Adds
2 SISO DFEs	$2\left(N_f P + N_b\right)$	$2\left(N_f P + N_b\right) - 2$
Hybrid DFE	$2\left(N_f P + N_b\right)$	$2\left(N_f P + N_b\right) - 2$
MIMO DFE 2×2	$4\left(N_f P + N_b\right)$	$4\left(N_f P + N_b\right) - 2$

Table 3.7 Computational Complexity of Combined Error-RLS Algorithm

Operation	Real Divides	Complex Multiplies	Complex Adds
$\boldsymbol{\Pi}(k)$ Update	1	$2L^2$	$2L(L-1)$
$\boldsymbol{K}(k)$ Update		$8L + 2$	$6L - 1$
$\boldsymbol{\epsilon}(k)$ Update		$2L$	$2L$
$\boldsymbol{w}_{\mathrm{CE}}(k)$ Update		$2L$	$2L$
$\boldsymbol{P}_{\mathrm{CE}}(k)$ Update		$9L^2/2$	$4L^2 - 2L$
Total	1	$13L^2/2 + 12L + 2$	$6L^2 + 6L - 1$

Error-RLS algorithm, which at first glance might appear to have higher complexity relative to the original RLS algorithm. Looking more closely at this algorithm, we see that the CE-RLS algorithm requires the inversion of a Hermitian 2×2 matrix, which is well known to be

$$\begin{pmatrix} a & b \\ b^* & c \end{pmatrix}^{-1} = \frac{1}{ac - |b|^2} \begin{pmatrix} c & -b \\ -b^* & a \end{pmatrix}.$$

Forming the right side of this equation requires 1 read divide, approximately 2 complex multiplies and 1 complex add. Thus, for each iteration the Combined Error-RLS algorithm requires the computations summarized in Table 3.7.

Finally, in Table 3.8 we summarize the total computational complexity for all of the DFEs we consider for a 2×1 TR-STBC system. We include both the complexity of calculating the equalizer output and of updating the tap weights using the adaptive equalization algorithms. For the independent SISO DFEs pairs we double a single SISO DFE's complexity to obtain the total complexity. The hybrid DFE complexity is calculated on a case by case basis.

Table 3.8 Total Computational Complexity of DFEs for 2×1 TR-STBC System

Architecture	Algorithm	Real Divs	Complex Mults	Complex Adds
2 SISO	LMS		$4L+1$	$4L$
Hybrid	CE-LMS		$4L+1$	$4L$
MIMO	MIMO LMS		$8L+1$	$8L$
2 SISO	NLMS	2	$4L+2$	$4L+3$
Hybrid	CE-NLMS	2	$4L+2$	$4L+3$
2 SISO	RLS	2	$7(L^2+L)$	$6L^2+2L-1$
Hybrid	AE-RLS	2	$13L^2/2+7L$	$6L^2+2L-1$
Hybrid	CE-RLS	1	$13L^2/2+12L+2$	$6L^2+6L-1$

Thus far, we have only tallied up the required additions, multiplications, and divisions to compute the equalizer output and to update the filter tap weights. These provide a numerical benchmark, but are by no means an absolute measure of complexity when it comes to actual implementation. In the next section we examine how matrix operations are implemented in hardware and the trade-offs between reducing hardware resources and increasing the computational latency.

3.4.3 Implementation Complexity Analysis

Matrix or vector multiplies are implemented in hardware using Multiply-ACcumulate (MAC) units, consisting of a multiplier and an accumulator which can act both as an adder and a subtractor. Each MAC performs a b-bit multiply-accumulate. Since both the data and channel are complex for our TR-STBC application, all of our calculations involve complex values.

Multiplying two complex numbers $A \triangleq a_r + j\,a_i$ and $B \triangleq b_r + j\,b_i$, we know that $AB = (a_r b_r - a_i b_i) + j\,(a_r b_i + a_i b_r)$. Thus, multiplication between two complex numbers requires 4 multiply and 2 addition operations (or 4 MACs) due to their real and imaginary components. Multiplication between a real and a complex number, on the other hand, only requires 2 multiplies (2 MACs) and zero additions since $a_r B = a_r b_r + j\,a_r b_i$.

For simplicity, we now analyze matrix and vector multiplies assuming real numbers, which only require 1 MAC per multiply. We define $n \times m$ matrix $\boldsymbol{A}$ and $m \times 1$ vector $\boldsymbol{b}$ and consider the following scenarios:

Complexity of multiplying a $n \times m$ matrix by a $m \times 1$ vector ($\boldsymbol{Ab}$):

- Solution 1: once-off full parallel multiply-accumulate: n rows of $\boldsymbol{A}$ simultaneously multiplied by column vector $\boldsymbol{b}$. Requires 1 MAC cycle, performing $n \times m$ MACs simultaneously.
- Solution 2: serial row multiply-accumulate: n rows of $\boldsymbol{A}$ sequentially multiplied by $\boldsymbol{b}$, accumulating the sum of products after every row multiply. Requires n MAC cycles, each time performing m MACs simultaneously.
- Solution 3: full serial multiply-accumulate: element by element multiply and accumulate across every row of $\boldsymbol{A}$, with a single row element in $\boldsymbol{b}$. Reuses a single multiplier. Requires $n \times m$ MAC cycles, each time performing a single MAC.

Complexity of multiplying a $n \times m$ matrix by its transpose ($\boldsymbol{AA}^T$):

- Solution 1: loop n times the once-off full parallel multiply-accumulate Solution 1 of above. Requires n MAC cycles, performing $n \times m$ MACs simultaneously.
- Solution 2: extension of Solution 2 of above, accumulate after each row on column operation for n columns in $\boldsymbol{A}^T$. Requires $n \times n$ MAC cycles, each time performing m MACs.
- Solution 3: extension of Solution 3 of above, due to replacement of single column vector $\boldsymbol{b}$ with $\boldsymbol{A}^T$ of n columns, requires $n^2 \times m$ MAC cycles, each time performing a single MAC.

A further level of customization is to consider the b-bits in each number within the matrices. In this case, each MAC mentioned previously can be:

- Fully serial (takes b clock cycles, uses a single adder)
- Fully parallel (takes 1 clock cycle, uses b adders)
- Anything in between these extremes.

As an example, we revisit the computations involved in forming matrix $\boldsymbol{R}_\delta$ of (3.10). Ignoring the block Toeplitz structure of $\boldsymbol{G}$, to form matrix multiplication $\boldsymbol{GG}^H$, Solution 1 would require a maximum of $n \times m = 25 \times 12 = 300$ MACs, and would take

$n = 25$ multiply and accumulate (MAC) cycles. However, taking into account the block Toeplitz structure of $\boldsymbol{G}$, we can equate $n = P$, thus reducing the overall computations to $P \times m = 5 \times 12 = 60$ MACs over $n = P = 5$ MAC cycles. For the simplified case, Solutions 2 and 3 would respectively only need $m = 12$ MACs over $n \times n = 25$ MAC cycles, and $m = 1$ MAC over $n^2 \times m = 300$ MAC cycles. Generally, it is desirable to conserve and reuse DSP blocks and multipliers. Thus, Solutions 2 or 3 would be the best choice. To compensate for the resulting latency, we could use an FPGA with a higher clock speed.

In conclusion, implementing a practical TR-STBC equalizer in real-time hardware involves multiple trade-offs. But, by using our computational complexity calculations for the various architectures and algorithms, an equalizer designer should be able to build an equalizer with adequate complexity to achieve good real-time performance. We are now ready to compare our DFE architectures, algorithms and analysis using Monte Carlo simulations of practical wireless channels.

3.5 COMPUTER SIMULATIONS

The first step in constructing a real-time TR-STBC equalization system is to use Monte Carlo computer simulations to study the floating point performance of the possible algorithms and architectures. In this section we use 64-bit floating point MATLAB simulations of complex baseband wireless channel models to investigate the dynamic and steady-state behaviors of our equalizers. We simulate the 2×1 TR-STBC architecture encoding QPSK modulated data symbols transmitted from two antennas and received by one antenna.

Wireless channels typically fall into two categories: fixed wireless channels where the transmitter and receiver are mounted at static locations, and mobile wireless channels where the transmitter and/or receiver move with time. For example, HDTV broadcasting and WLANs operate over fixed or approximately fixed wireless channels, whereas cellular telephone systems operate over mobile wireless channels. We therefore choose to simulate channel models from each of these categories. Our fixed channel model, which we denote the FIXED CHANNEL, is based on experimental measurements of

an indoor quasi-static channel captured using the real-time MIMO hardware platform designed by the Tait Electronics STAR group [11]. Our mobile channel model, which we denote the MOBILE CHANNEL, is the standard ITU Vehicular A channel model [40].

This simulation section is organized as follows. We first introduce the details of the wireless channel models in Sec. 3.5.1. We then define the TR-STBC system and the equalizer parameters in Sec. 3.5.2. We begin our simulation study in Sec. 3.5.3, comparing our theoretical analysis to the simulated results for the three DFE architectures, as well as their dynamic convergence behaviors. In Sec. 3.5.4 we demonstrate the effects of channel estimation on the MSE performances and show the advantages and disadvantages of our new packet structure for high mobility channels. We investigate the ability of the Kick-start Method of Steepest Descent to increase the MSE convergence in Sec. 3.5.5. We then examine the dynamic convergence behavior of the NLMS and RLS based algorithms in Sec. 3.5.6. Finally, we compare the steady-state performances of the DFEs for both the Prototype and Brute Force packet structures in Sec. 3.5.7.

3.5.1 Wireless Channel Models

In this section we give details of our two channel models, the FIXED CHANNEL and the MOBILE CHANNEL. For both channel models we assume that the two transmit antennas either use polarizations that are perpendicular to each other or are spaced far enough apart so that their output streams are spatially uncorrelated. Therefore, we can build up each 2×1 MISO channel model using two independent realizations of a SISO channel. As mentioned in Sec. 2.2, we also assume that both dispersive channels are linear and can be modelled in discrete time using finite impulse response (FIR) filters with either fixed or time-varying filter tap weights. We finally assume that the observations at the receiver include additive white Gaussian noise which is independent of the transmitted symbols.

The FIXED CHANNEL uses the channel estimates of an actual indoor wireless channel captured using the real-time MIMO hardware platform designed by the Tait

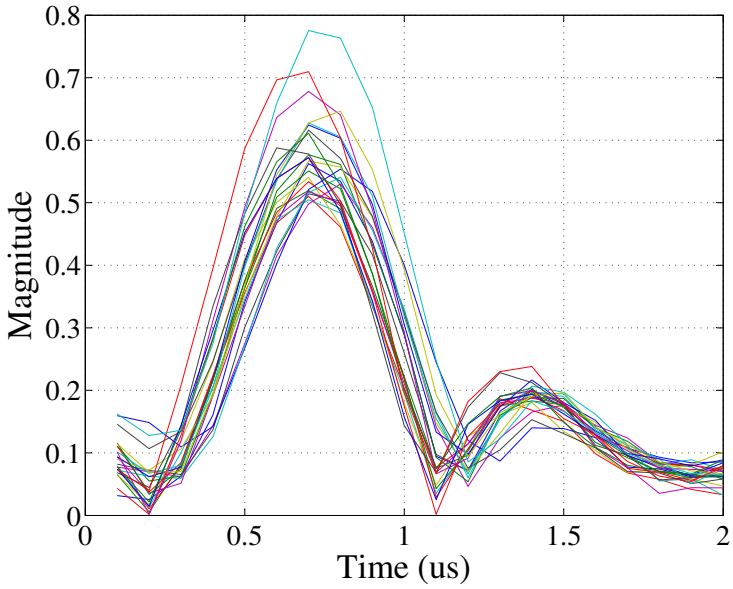

Figure 3.7 FIXED CHANNEL: Captured indoor wireless channel estimates of $|h_1(t)|$ over 25 data blocks ($v_h = 4$, $P = 5$)

Electronics STAR group [11]. For the measurements the transmitter and receiver were placed 30m apart, with one transmit antenna using horizontal polarization and the other using vertical polarization. All of the antennas were of the Patch Inverted "F" Antenna (PIFA) type. The specific system parameter values, including the length of the channel estimation sequence and the sampling frequency, are given in Sec. 3.5.2.

In Fig. 3.7 we show the magnitudes of 25 different CIR estimates for the channel between one particular transmit antenna and the receive antenna based on 25 TR-STBC blocks. The CIRs are the combined baseband digital-to-baseband digital responses of the transceiver and channel, including the pulse-shaping filters, the mixers, the analog front ends, and the physical wireless channel. From Fig. 3.7 we see that for this fixed wireless scenario the CIR estimates vary only slightly from block to block. For the other transmit antenna, we have found this to be the case as well. Therefore, for our FIXED CHANNEL computer simulations, we have arbitrarily chosen the two CIRs shown in Fig. 3.8 from the 17th TR-STBC block, corresponding to the two transmit antennas. We hold the two CIRs fixed for all of the TR-STBC blocks and add white Gaussian noise to each block.

The MOBILE CHANNEL is generated using the standard ITU Vehicular A high

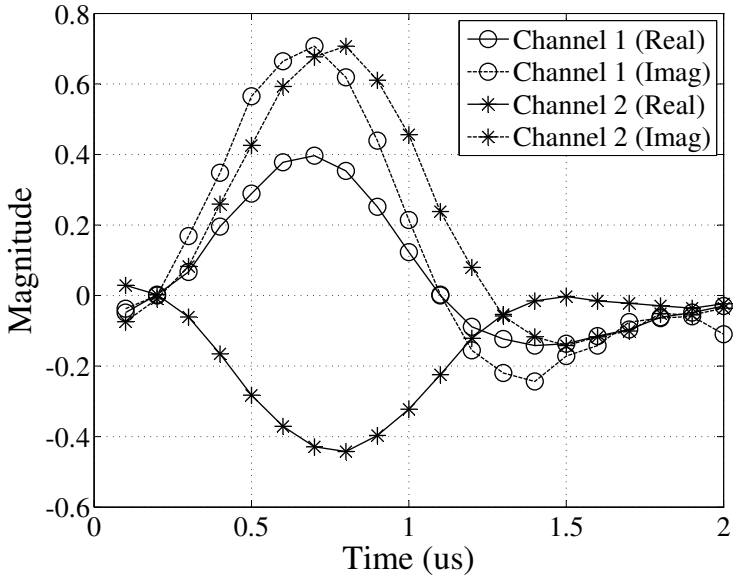

Figure 3.8 FIXED CHANNEL: Complex baseband CIRs $h_1(t)$ and $h_2(t)$ for TR-STBC block 17 ($v_h = 4$, $P = 5$)

Table 3.9 ITU Vehicular A channel tap weight settings. Each tap weight has fading with classic Doppler spectrum.

Tap Weight	Relative Delay (ns)	Average Power (dB)
1	0	0.0
2	310	-1.0
3	710	-9.0
4	1090	-10.0
5	1730	-15.0
6	2510	-20.0

mobility channel model [40]. To simulate this time-varying channel, we use an FIR filter with time-varying tap weights. The power delay profile of this channel is summarized in Table 3.9. It includes 6 multipath rays and has an RMS delay spread of 370ns. The Doppler spectrum of each tap weight, based on the classic Jakes' model [41], is generated using a modified version of Smith's algorithm [42]. In our simulations, we make the assumption that the maximum vehicle speed is constant over all of the TR-STBC blocks, but for some scenarios we compute our results for a variety of vehicle speeds.

Figure 3.9 shows example sample paths of the six independent Rayleigh fading rays

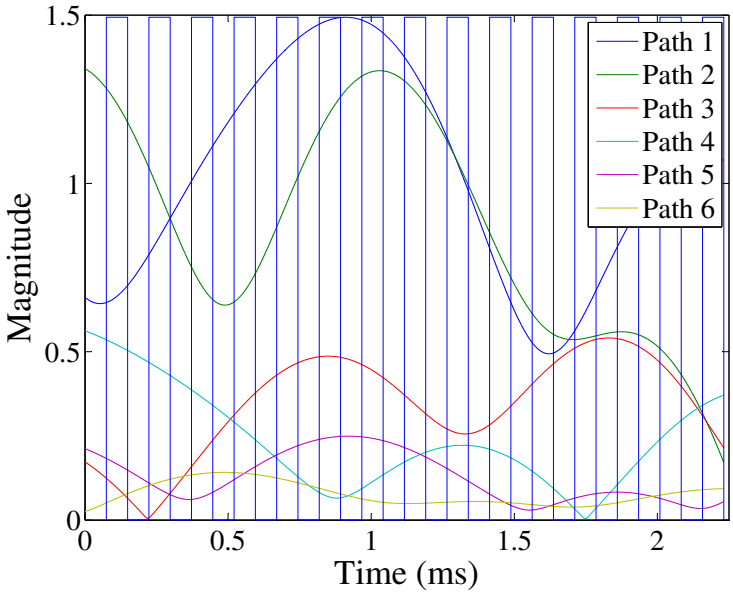

Figure 3.9 MOBILE CHANNEL: ITU Vehicular A channel over 30 data packets ($v_h = 26$, $P = 7$). The vertical lines indicate the packet boundaries.

over 30 consecutive data packets. Here we use a carrier frequency of 5GHz and a Doppler frequency of 555Hz, corresponding to a maximum vehicle speed of 120km/hr. Within each packet we see that the channel tap weights do not vary drastically, making the pseudo-static channel assumptions in this thesis reasonable. However, care will need to be taken in forming estimates of the CIRs for each packet.

3.5.2 TR-STBC System and Equalizer Parameters

For the FIXED CHANNEL, the transceiver system parameters are summarized in Table 3.10. We tailor our simulation parameters to that of the STAR hardware platform [11] to enable us to utilize the captured channel data from the platform. This also facilitates the re-use of components within the existing hardware/RF platform in the real-time implementation of our new DFE architectures and algorithms. Information on other aspects of the platform can be found in the STAR's performance specifications [43].

In our simulations we use a CIR length $L_h = 4$ and guard length $L_g = 5$, complying with the STAR hardware specification [44]. For the Brute Force packet structure of

Table 3.10 FIXED CHANNEL: System Parameters

Parameters	**Values**
Symbol Rate	2 Mbaud
Pulse Shaping Filters	2MHz BW Root Raised Cosine ($\alpha = 0.5$)
Carrier Frequency	2.4 GHz
Intermediate Frequency (IF)	15 MHz
Oversampling Rate P	5
Feedforward Filter Length N_f	5 Symbols
Feedback Filter Length N_b	5 Symbols

Table 3.11 MOBILE CHANNEL: System Parameters

Parameters	**Values**
Symbol Rate	10 Mbaud
Carrier Frequency	5 GHz
Oversampling Rate P	7
Feedforward Filter Length	4 Symbols
Feedback Filter Length	2 Symbols
Vehicle Speed (Default Value)	120 km/hr

Fig. 3.4, to be compatible with the original Viterbi equalizer hardware, we let $L_{teq} = 0$, $L_{tce} = 25$, $L_d = 55$, and choose a total packet length of $L_p = 150$. For the Prototype structure of Fig. 3.6, we set $L_{teq} = L_{tce} = 10$ and $L_d = 55$, maintaining a total packet length of $L_p = 150$ for a fair comparison with the Brute Force packet structure.

Similarly, the system parameters for the MOBILE CHANNEL are shown in Table 3.11. Unless otherwise stated, the vehicle speed is set to 120km/hr, resulting in the maximum Doppler frequency. For the MOBILE CHANNEL we have selected the following packet parameter values: CIR length $L_h = 26$, guard length $L_g = 26$, data load length $L_d = 256$ and total packet length $L_p = 744$. Both the Prototype and Brute Force packet section lengths are summarized in Table 3.12. The Brute Force packet structure was optimized through simulation.

3.5.3 LMS Dynamic Convergence Behavior for Three DFE Architectures

We are now ready to consider the simulated performance of our TR-STBC equalizers using the two channel models. In this section we simulate the three DFE architectures of Sec. 3.1 running the various LMS algorithms, comparing their dynamic behaviors

Table 3.12 MOBILE CHANNEL: Prototype vs. Brute Force TR-STBC Packet Section Lengths

Packet Section Lengths	L_{tce}	L_{teq}	L_g	L_d	L_p
Prototype	64	64	26	256	$L_{tce} + L_{teq} + 4L_g + 2L_d = 744$
Brute Force	128	26	26	256	$L_{tce} + L_{teq} + 3L_g + 2L_d = 744$

with each other and with our statistical convergence analysis introduced in Sec. 2.4. The first half on this section is dedicated to analyzing the results for the FIXED CHANNEL, followed by the MOBILE CHANNEL in the second half of the section.

For both channel models, to isolate the effects of the different DFE structures on the dynamic convergence behavior of the equalizers, in this section all of the Monte Carlo simulations are carried out for the case of perfect knowledge of the CSI by the TR-STBC decoder. Also, the equalizer is in training mode for the whole packet length (unlike the Brute Force and Prototype packet structures). The equalizer taps are initialized to zero for each new packet, and the results are averaged over 100 packets. The LMS step-size is identical for all three DFE structures.

For the FIXED CHANNEL, the step-size is set to $\mu = 0.00025$ for SNR $\leq$ 10dB and $\mu = 0.0005$ for SNR $>$ 10dB. First considering the SISO DFE, we compare the simulated MSE trajectory with the statistical MSE trajectories for SNRs of 10, 20, and 30dB in Figs. 3.10 – 3.12. We observe from Figs. 3.10 – 3.12 that the statistical MSE trajectories align closely with the simulated MSE trajectory for all three SNRs, with the best-fit statistical line generated using Independence Theory. We also observe that the Independence Theory trajectory consistently has the highest MSE, while the Steepest Descent trajectory is overly optimistic, ignoring the noisy estimates of the gradient.

We next consider the Novel Hybrid DFE, comparing the simulated MSE trajectory with the analytical MSE trajectories for the FIXED CHANNEL in Figs. 3.13 – 3.15. In this case, the Hybrid DFE's simulated MSE trajectory has closest alignment with the Small Step-size MSE trajectory, situated between the Independence Method and the Steepest Descent trajectories. The Independence Theory curve again has the highest MSE. With increasing SNR the Small Step-size Theory trajectory becomes closer to

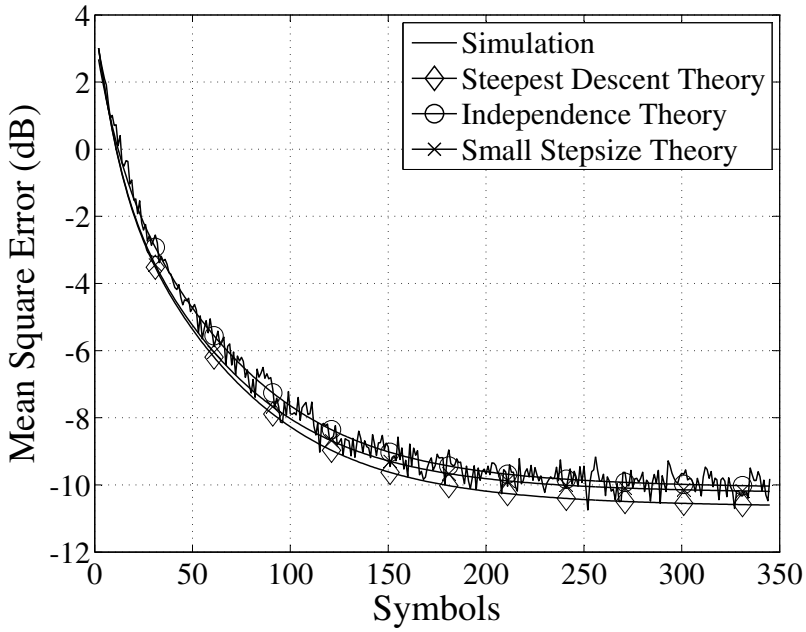

Figure 3.10 FIXED CHANNEL: MSE trajectory for SISO DFE (10dB SNR)

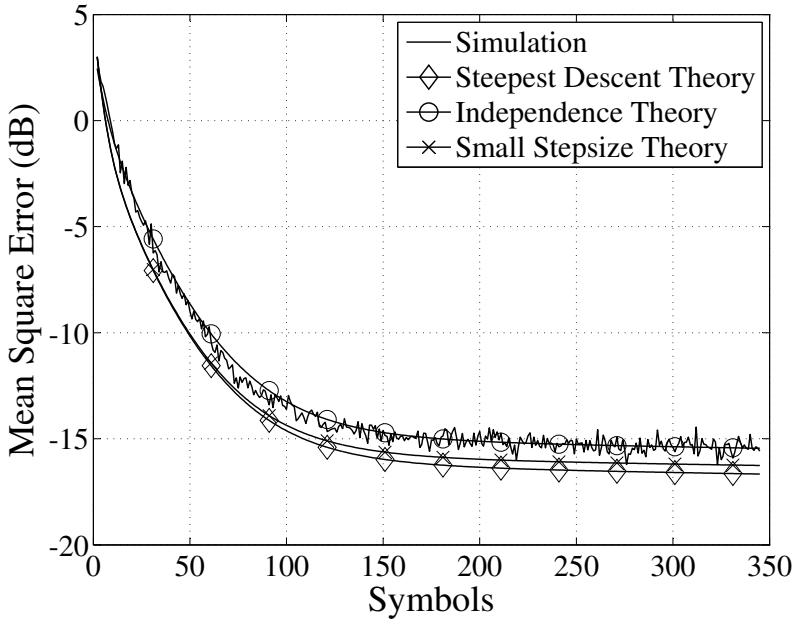

Figure 3.11 FIXED CHANNEL: MSE trajectory for SISO DFE (20dB SNR)

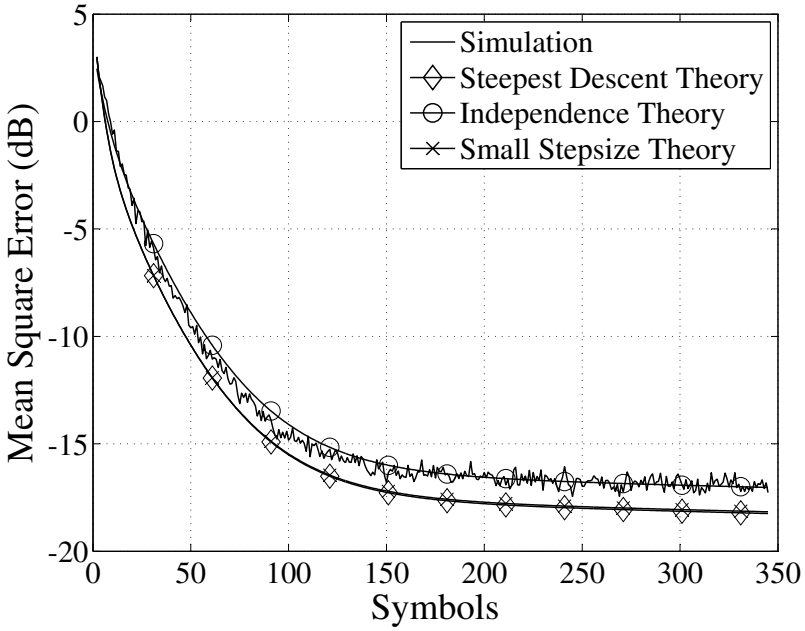

Figure 3.12 FIXED CHANNEL: MSE trajectory for SISO DFE (30dB SNR)

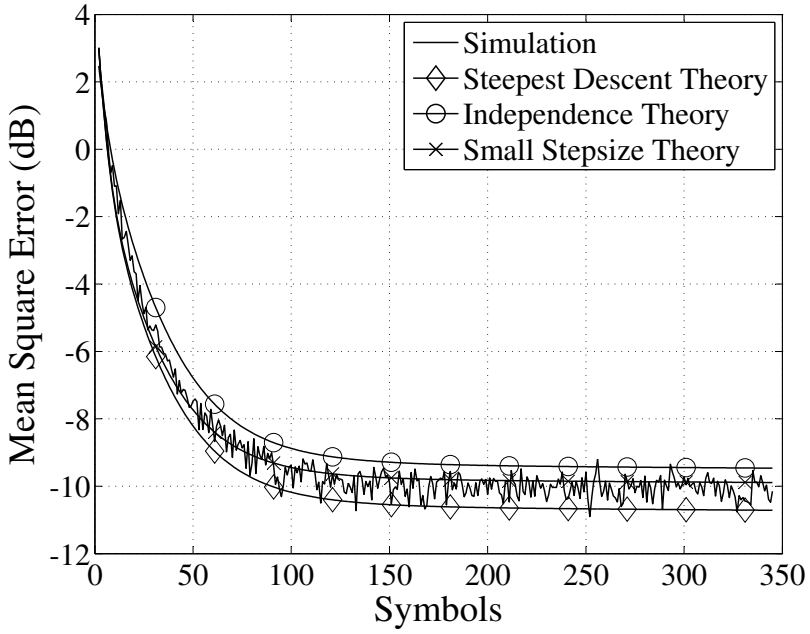

Figure 3.13 FIXED CHANNEL: MSE trajectory for Novel Hybrid DFE (10dB SNR)

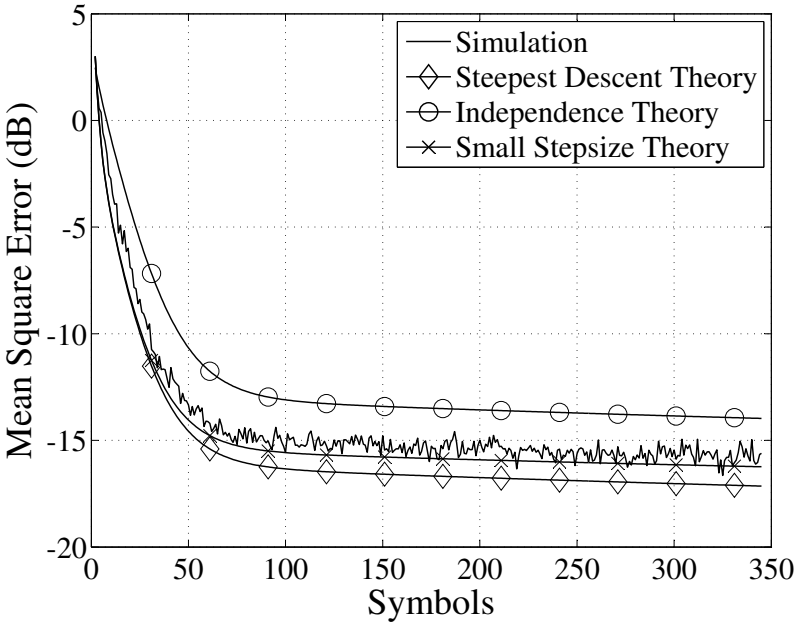

Figure 3.14 FIXED CHANNEL: MSE trajectory for Novel Hybrid DFE (20dB SNR)

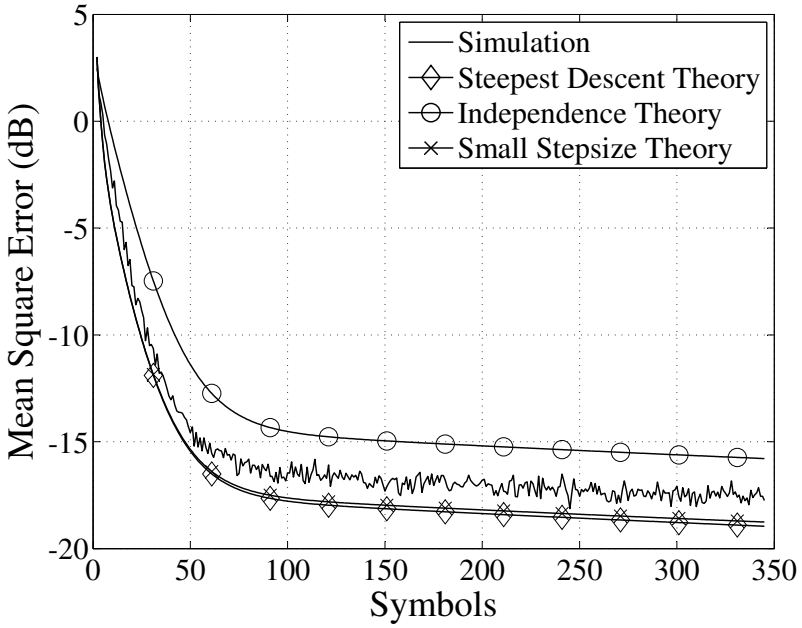

Figure 3.15 FIXED CHANNEL: MSE trajectory for Novel Hybrid DFE (30dB SNR)

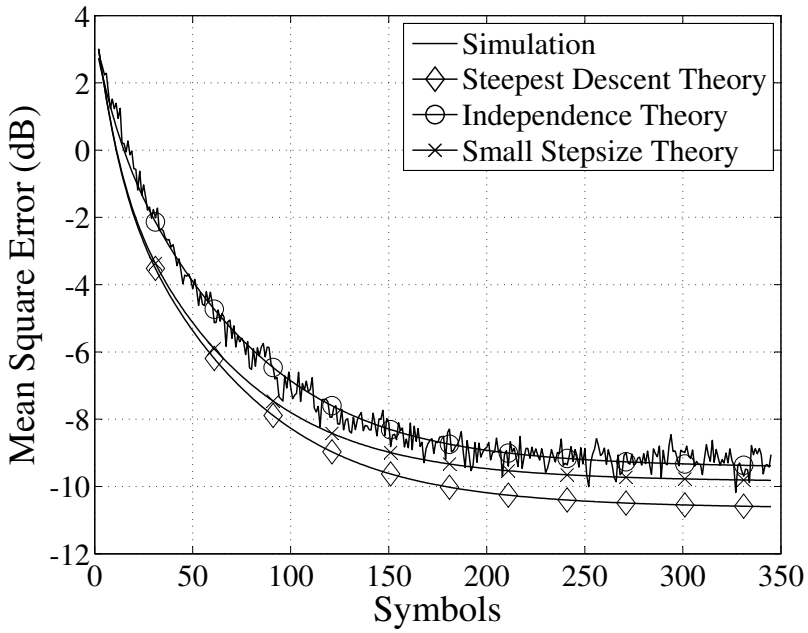

Figure 3.16 FIXED CHANNEL: MSE trajectory for MIMO DFE (10dB SNR)

the Steepest Descent one. Again, the Steepest Descent Method yields the lowest MSE trajectory for all SNRs.

Finally, we study the MIMO DFE, comparing its simulated and theoretical MSE trajectories in Figs. 3.16 – 3.18. We once again find the Independence Theory, Small Step-size Theory and Steepest Descent Theory curves in the order of highest to lowest MSE trajectories in Figs. 3.16 – 3.18. For the MIMO DFE, the simulated MSE curve is aligned most closely with the Independence Theory for all SNR values.

Thus far, our results have shown that the two alternative statistical convergence theories yield better approximations to the simulated MSE trajectory than the overly optimistic Steepest Descent Theory. Also, for the FIXED CHANNEL the MIMO DFE delivers the worst MSE performance while taking up twice the complexity of the Parallel SISO and Novel Hybrid DFEs.

We now proceed to the MOBILE CHANNEL, setting the step-size to $\mu = 0.002$ for all SNRs. For the SISO DFE, in Figs. 3.19 – 3.21 we show the simulated and analytical MSE trajectories. We observe that all of the MSE curves line up closely, confirming the agreement between simulation and theory. We also observe that the Independence Method yields the highest MSE, while the Small Step-size curve becomes closer to the

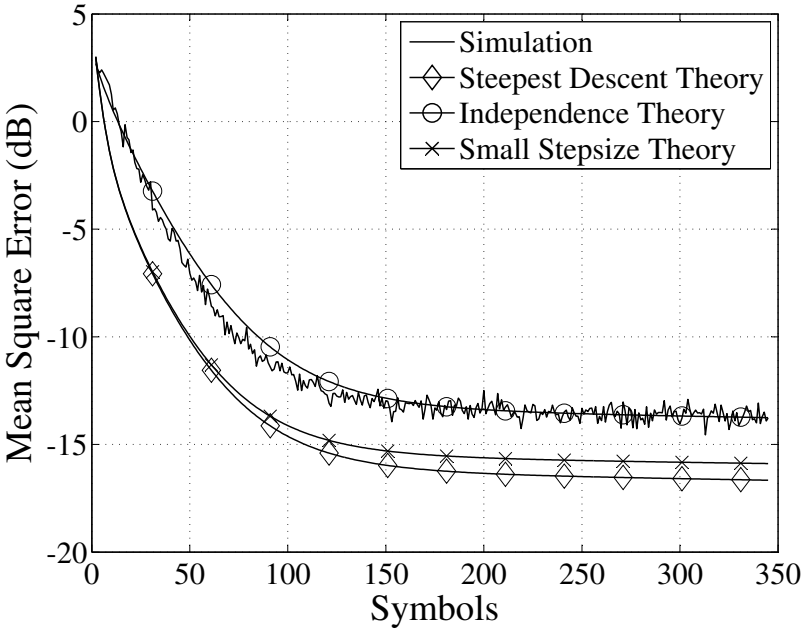

Figure 3.17 FIXED CHANNEL: MSE trajectory for MIMO DFE (20dB SNR)

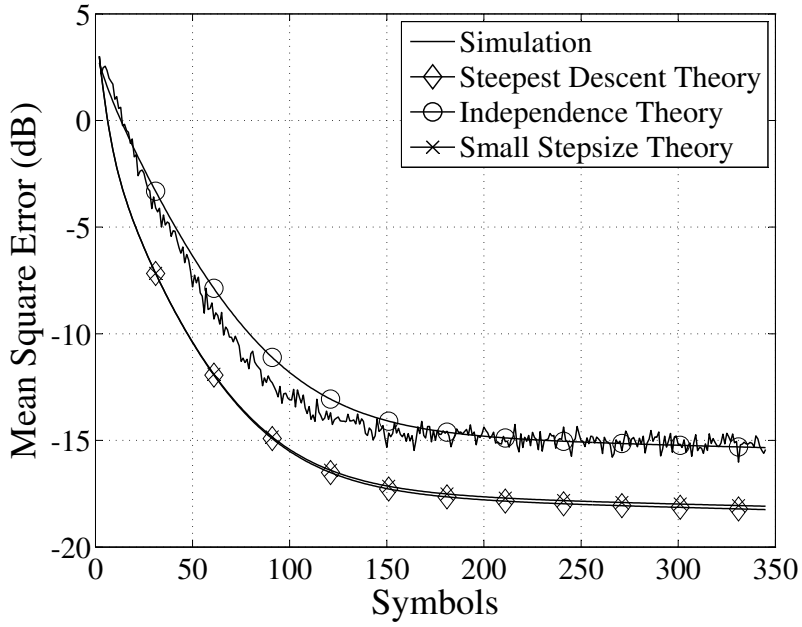

Figure 3.18 FIXED CHANNEL: MSE trajectory for MIMO DFE (30dB SNR)

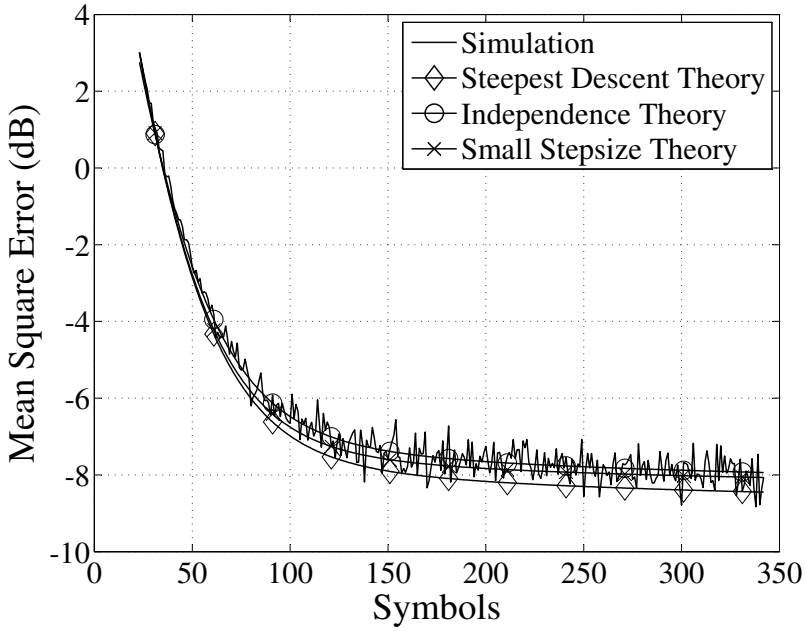

Figure 3.19 MOBILE CHANNEL: MSE trajectory for SISO DFE (10dB SNR)

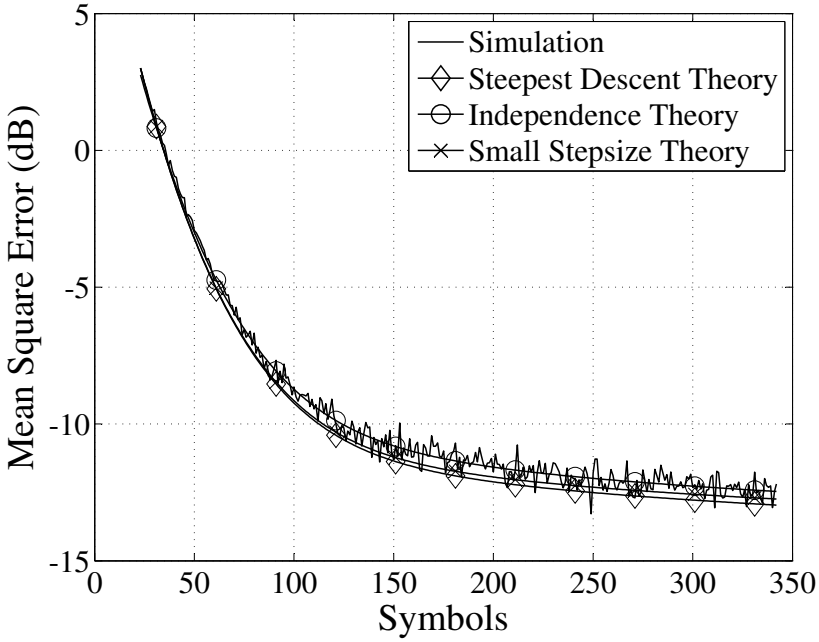

Figure 3.20 MOBILE CHANNEL: MSE trajectory for SISO DFE (20dB SNR)

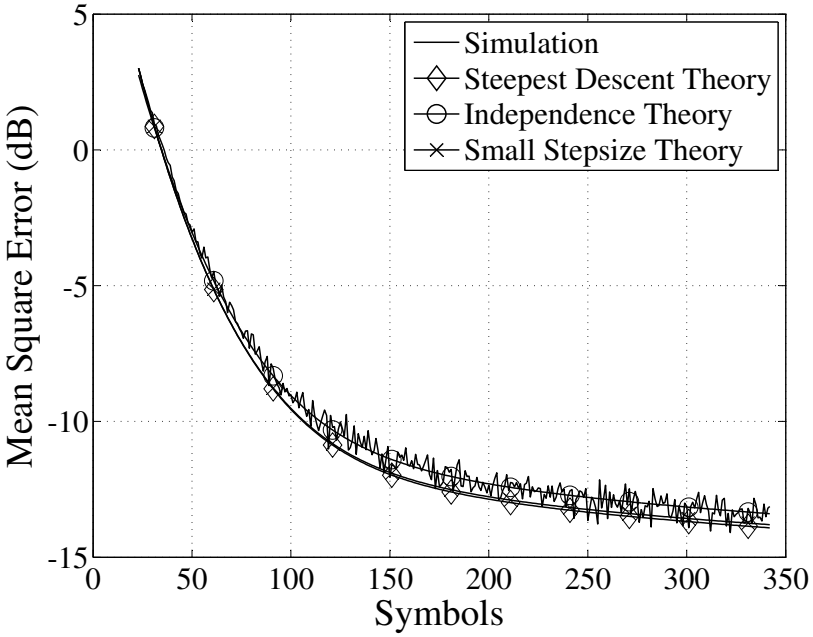

Figure 3.21 MOBILE CHANNEL: MSE trajectory for SISO DFE (30dB SNR)

Steepest Descent one with increasing SNR.

For the Novel Hybrid DFE we plot the simulated and theoretical MSE trajectories in Figs. 3.22 – 3.24. We see that for the Hybrid DFE the simulated MSE trajectory is aligned most closely with the Small Step-size trajectory, situated between the Independence Method and the Steepest Descent trajectories. This is similar to the Hybrid DFE's earlier results for the FIXED CHANNEL in Figs. 3.13 – 3.15.

Finally, for the MIMO DFE we draw the simulated and analytical MSE trajectories in Figs. 3.25 – 3.27. These figures show a clear gap between the Independence Method and the Small step-size/Steepest Descent trajectories. As with the FIXED CHANNEL case, the MIMO DFE's simulated MSE trajectory lies closest to the Independence Method curve for all SNRs. We once again find the Independence Theory, Small Step-size Theory and Steepest Descent Theory curves in the order of highest to lowest MSE trajectory.

In summary, using Monte Carlo simulations of the dynamic LMS convergence behaviors for both the MOBILE CHANNEL and the FIXED CHANNEL, we have demonstrated that the two alternative statistical convergence theories yield better MSE convergence approximations to the simulated MSE trajectory than the original Steepest

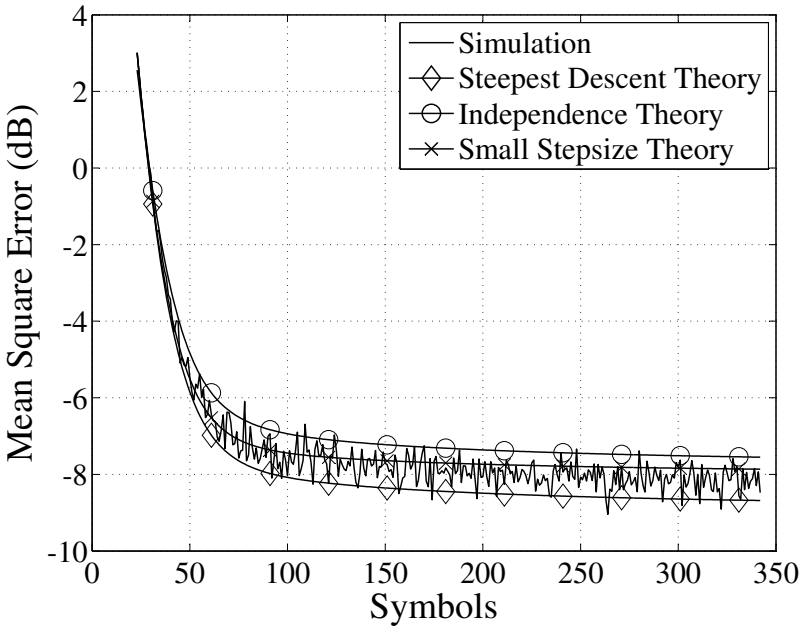

Figure 3.22 MOBILE CHANNEL: MSE trajectory for Novel Hybrid DFE (10dB SNR)

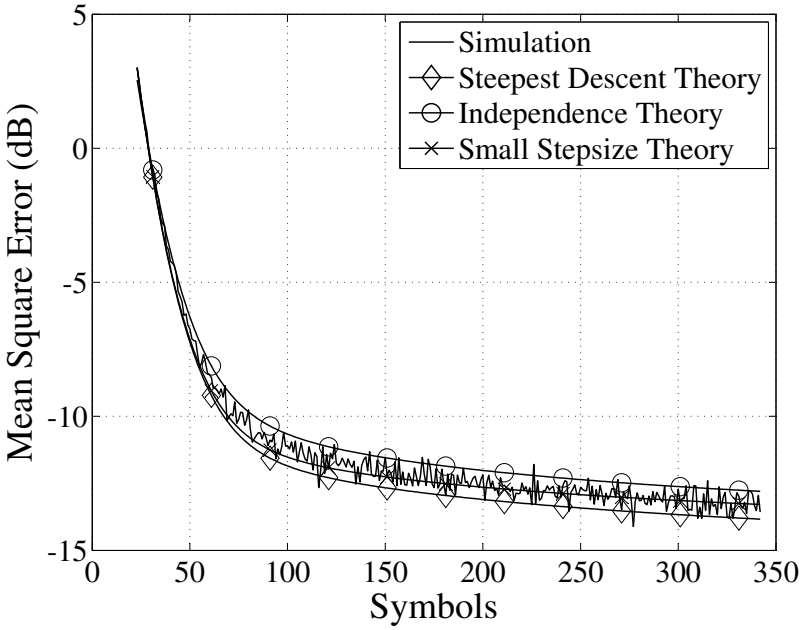

Figure 3.23 MOBILE CHANNEL: MSE trajectory for Novel Hybrid DFE (20dB SNR)

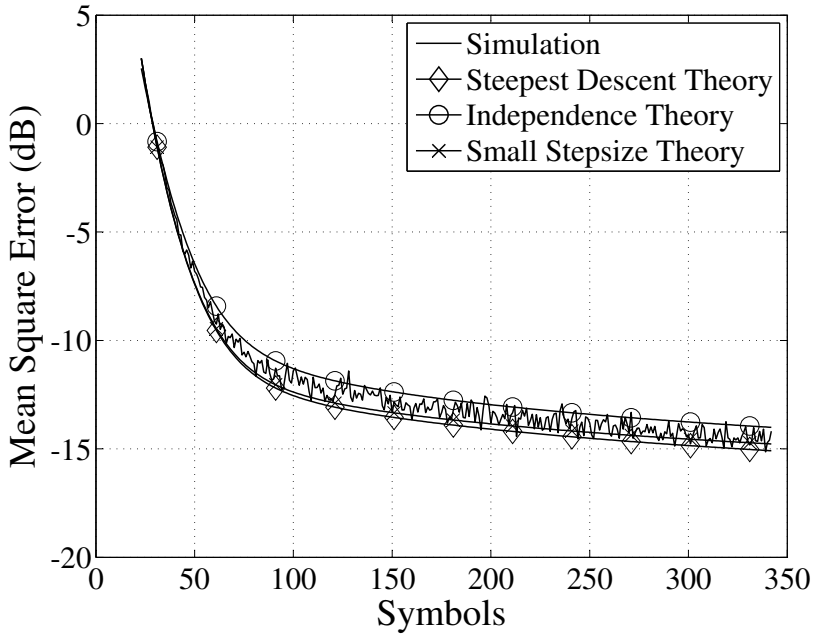

Figure 3.24 MOBILE CHANNEL: MSE trajectory for Novel Hybrid DFE (30dB SNR)

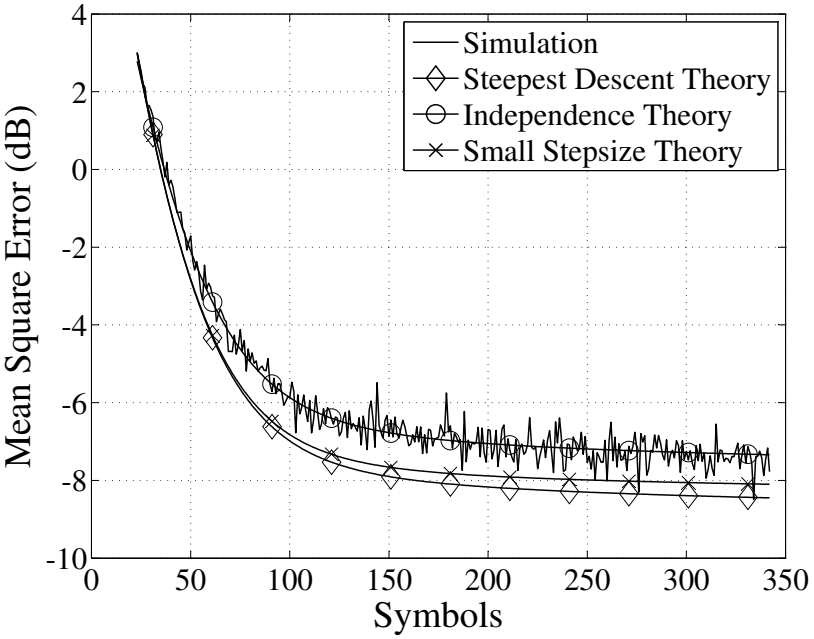

Figure 3.25 MOBILE CHANNEL: MSE trajectory for MIMO DFE (10dB SNR)

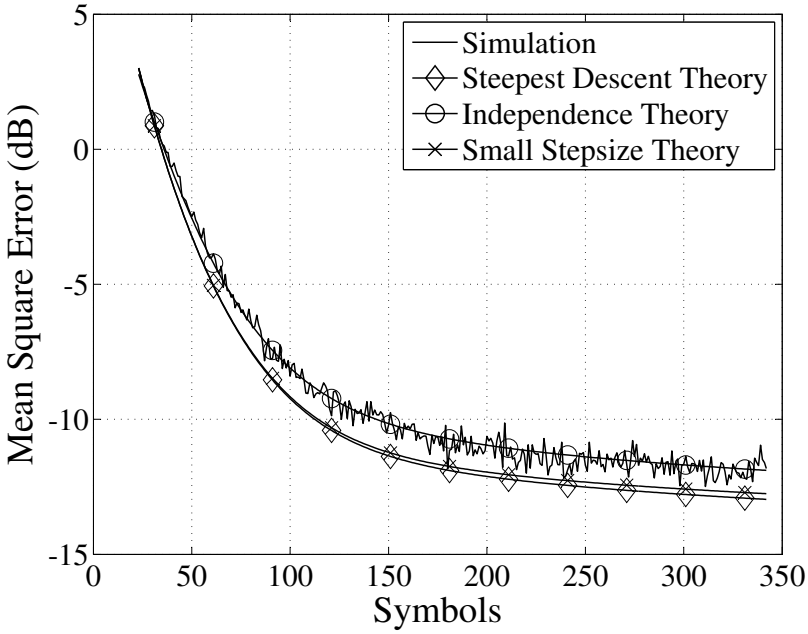

Figure 3.26 MOBILE CHANNEL: MSE trajectory for MIMO DFE (20dB SNR)

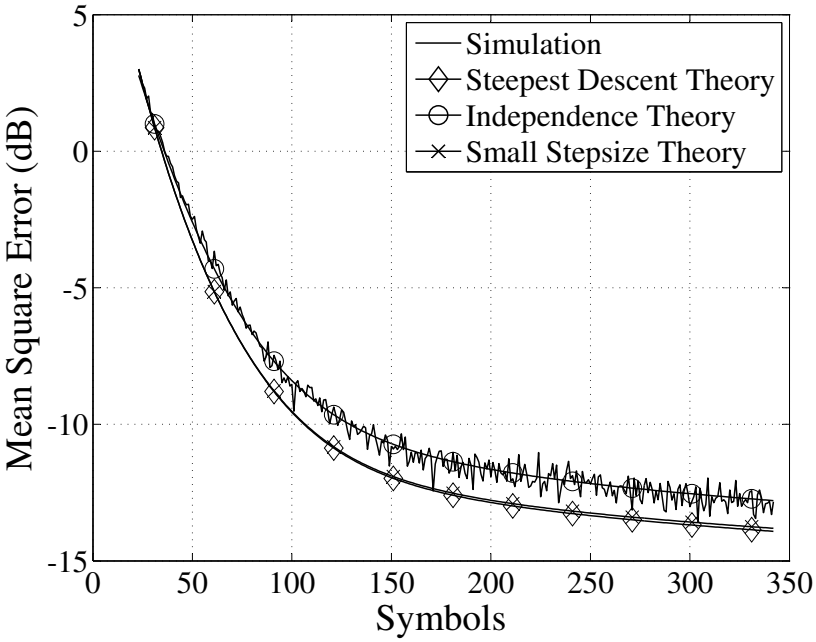

Figure 3.27 MOBILE CHANNEL: MSE trajectory for MIMO DFE (30dB SNR)

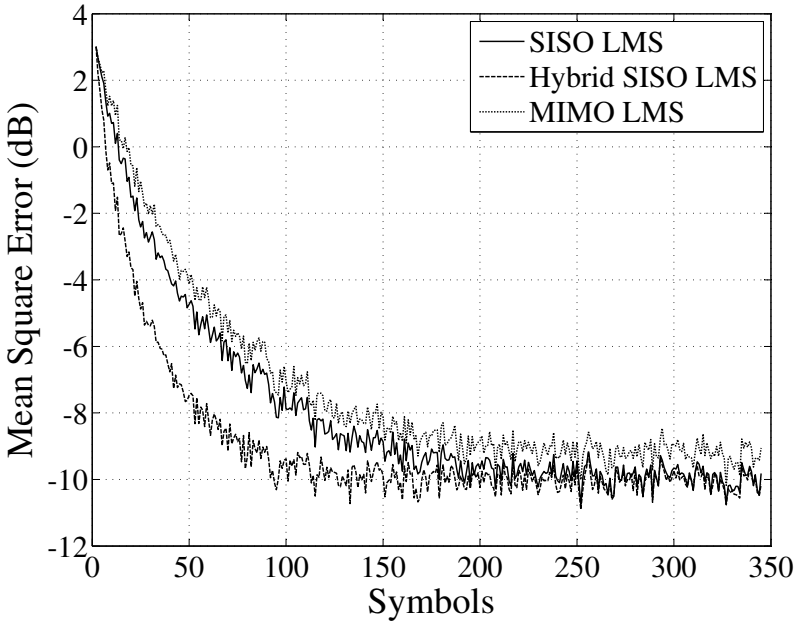

Figure 3.28 FIXED CHANNEL: DFE architecture comparison (10dB SNR)

Descent Theory. We also conclude that the relationships between the simulated MSE trajectory and the various statistical MSE trajectories differ for each DFE architecture, but remain consistent for both the FIXED and MOBILE CHANNELS. Finally, comparing the steady-state MSE values of all of the figures, we observe that the Hybrid LMS algorithm yields the best equalizer performance for a given complexity.

To see this explicitly, we finally compare the simulated MSE trajectories of all three DFE architectures from Sec. 3.1, first for the FIXED CHANNEL and then for the MOBILE CHANNEL. The results for the FIXED CHANNEL are shown in Figs. 3.28 – 3.30 for SNRs of 10, 20 and 30dB. For a 10dB SNR we see that the Novel Hybrid DFE produces the fastest convergence to steady-state, yielding an MSE margin of around 2dB over both the SISO and MIMO DFEs after the first 100 symbols. This MSE performance gap widens with increasing SNR, reaching 5dBs at 30dB SNR.

Similarly, comparisons of the simulated MSE trajectories of the three DFE architectures for the MOBILE CHANNEL are shown Figs. 3.31 – 3.33 for 10, 20 and 30dB SNRs. Again, we find that the Novel Hybrid DFE produces the fastest convergence to steady-state, while the MIMO DFE yields the slowest convergence as well as the highest steady-state MSE. The Hybrid DFE yields a maximum MSE margin of 3dB and

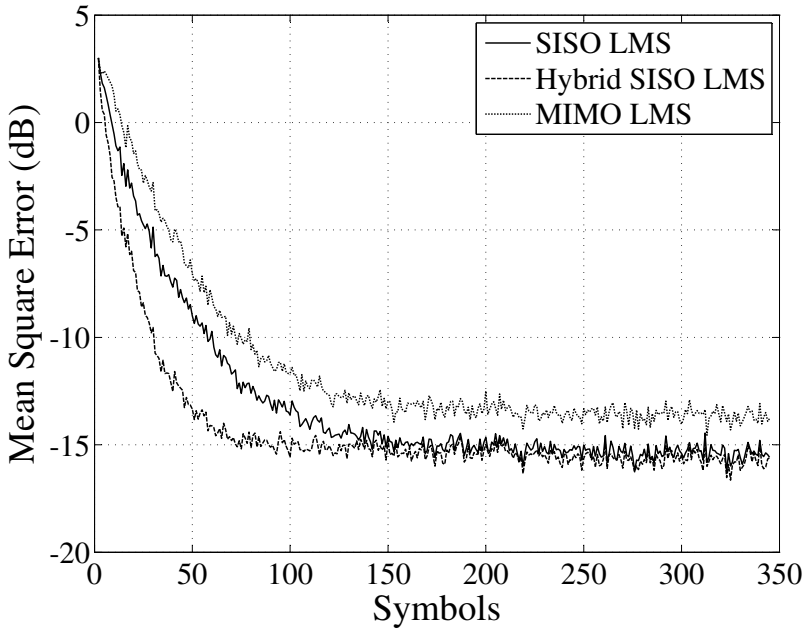

Figure 3.29 FIXED CHANNEL: DFE architecture comparison (20dB SNR)

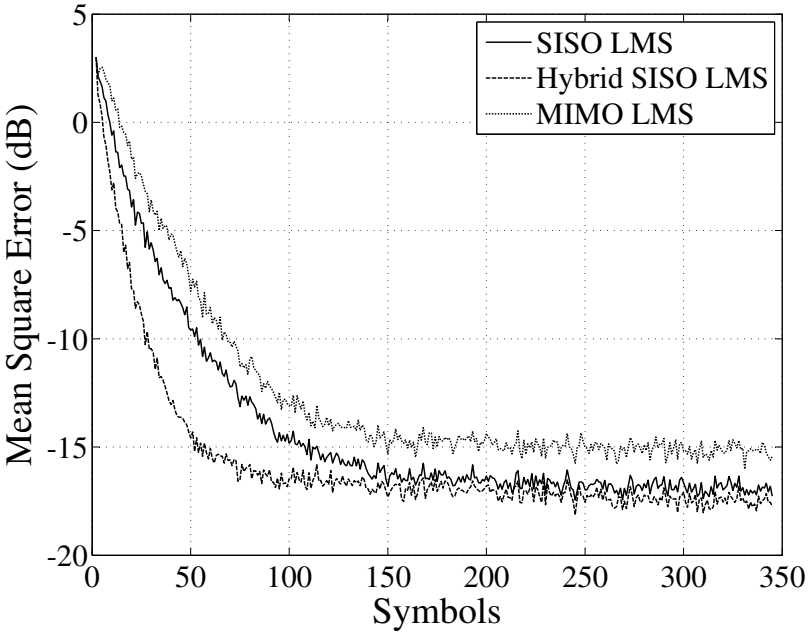

Figure 3.30 FIXED CHANNEL: DFE architecture comparison (30dB SNR)

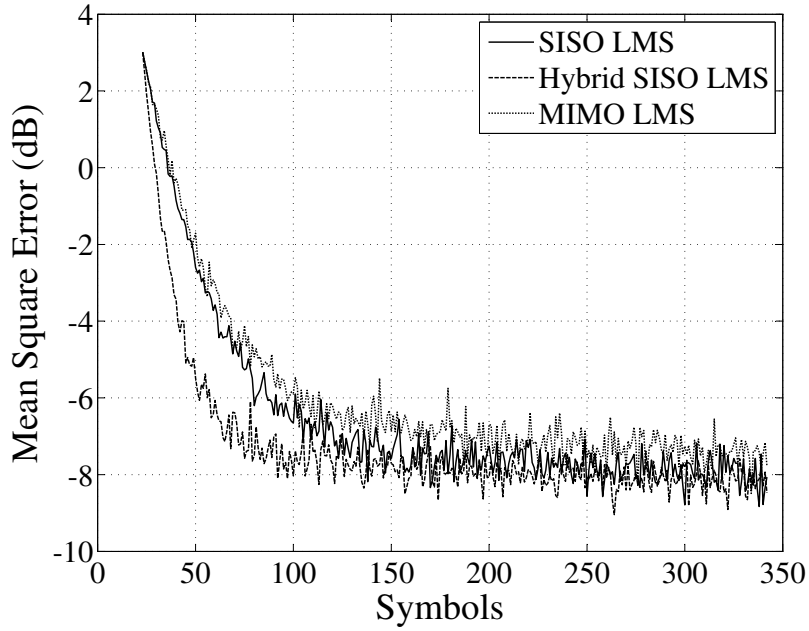

Figure 3.31 MOBILE CHANNEL: DFE architecture comparison (10dB SNR)

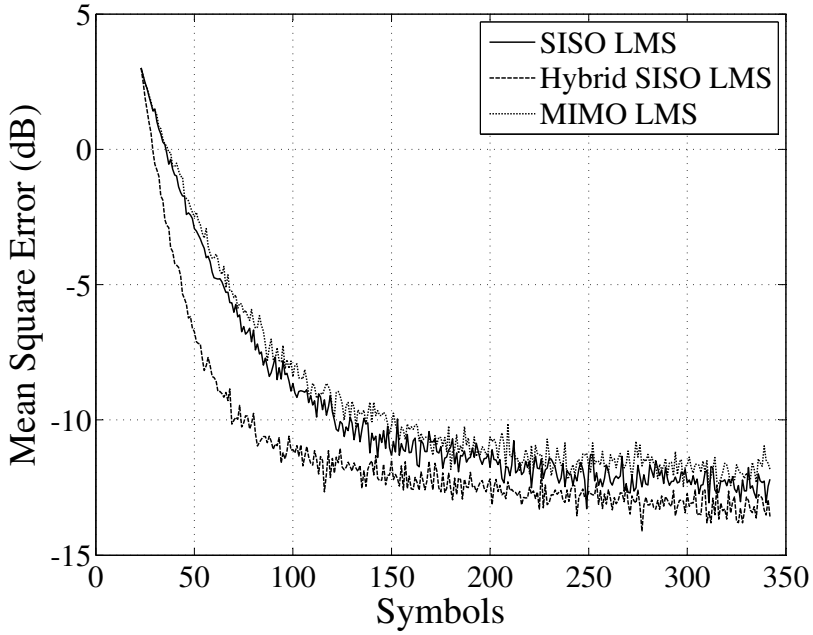

Figure 3.32 MOBILE CHANNEL: DFE architecture comparison (20dB SNR)

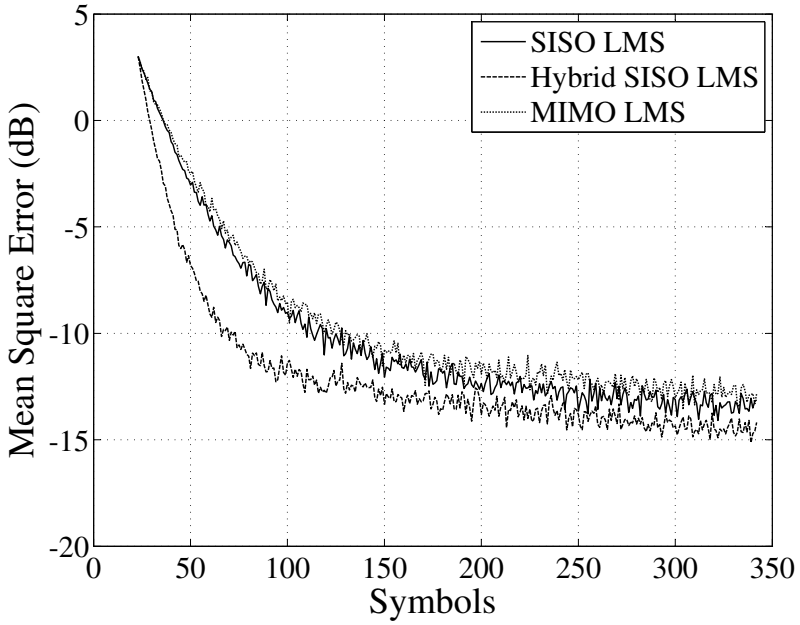

Figure 3.33 MOBILE CHANNEL: DFE architecture comparison (30dB SNR)

4dB, respectively, over the SISO and MIMO DFEs after 50-100 symbols. This margin widens with increasing SNR, reaching MSE gaps of 5dB over the SISO and MIMO DFEs for 20dB and 30dB SNRs. Thus, given its reasonable computational complexity and good performance for the two channel models, we recommend our Novel Hybrid DFE over the other two architectures.

In summary, the Hybrid DFE running the CE-LMS algorithm yields the fastest convergence with the lowest steady-state MSE for the least complexity, while the MIMO DFE has the slowest convergence with the highest steady-state MSE for the most complexity. We recommend the CE-LMS algorithm as the best candidate for replacing the conventional LMS algorithm in a DFE for a practical TR-STBC system. Finally, we conclude that the poor performance of the MIMO DFE for the LMS algorithm shows that it is not suitable for implementation in a TR-STBC system. Therefore, we do not consider the MIMO DFE any further in Sec. 3.5.

3.5.4 Effect of Packet Structure on MSE Performance

With our goal of practical implementation, we are especially interested in the effect of the various packet structures presented in Sec. 3.2 on the TR-STBC system's MSE

performance. Choosing the proper length and placement of the training sequence in the data block is particularly important for the time-varying MOBILE CHANNEL. The structure of the packet controls the trade-off between channel estimation and training of the equalizer, which respectively affect the accuracy of the TR-STBC decoder and the equalizer.

Here we simulate the MOBILE CHANNEL, averaging over 100 packets for the Novel Hybrid DFE at an SNR of 10dB. We compare the Brute Force data packet structure with the Prototype data packet structure using their MSE convergence performances. The equalizer taps are initialized to zero for each new packet. The LMS step-size is set to $\mu = 0.002$. We observe from Fig. 3.9 that the MOBILE CHANNEL varies continuously over consecutive data packets. Therefore, instead of re-initializing the equalizer tap weights to zero at the start of each new packet, we use the final tap weights from the previous packet as the starting tap weights.

We compare the MSE trajectories of the Brute Force packet structure and the Prototype packet structure with the case of perfectly known CSI and full packet training ("Perfect CSI Structure") in Fig. 3.34. The Perfect CSI curve of Fig. 3.34 is simulated using the same transceiver parameters and channels as those in Fig. 3.22. Because we do not re-initialize the tap weight to zero, the Perfect CSI curve in Fig 3.34 starts off 9dB lower than its analogous curve in Fig. 3.22. Also, the simulated MSE trajectory in Fig. 3.34 converges to a steady-state MSE which is 0.5dB lower than that of Fig. 3.22. The simulated MSE in Fig. 3.34 reaches steady state after 25 symbols as opposed to more than 50 symbols in Fig. 3.22. Thus, we conclude that exploiting the final equalizer tap weights for the subsequent packet in a time-varying channel offers significant equalization performance enhancement.

The Perfect CSI packet structure offers an upper bound on practical equalization performance, since in practice we cannot stay in training mode for the whole duration of the packet or have perfect knowledge of the TR-STBC channel at the receiver. Comparing the two "practical" packet structure implementations against each other, we see that the Brute Force and Prototype structures have similar initial MSE performances. The MSE trajectory for the Prototype packet structure dips into a deep null

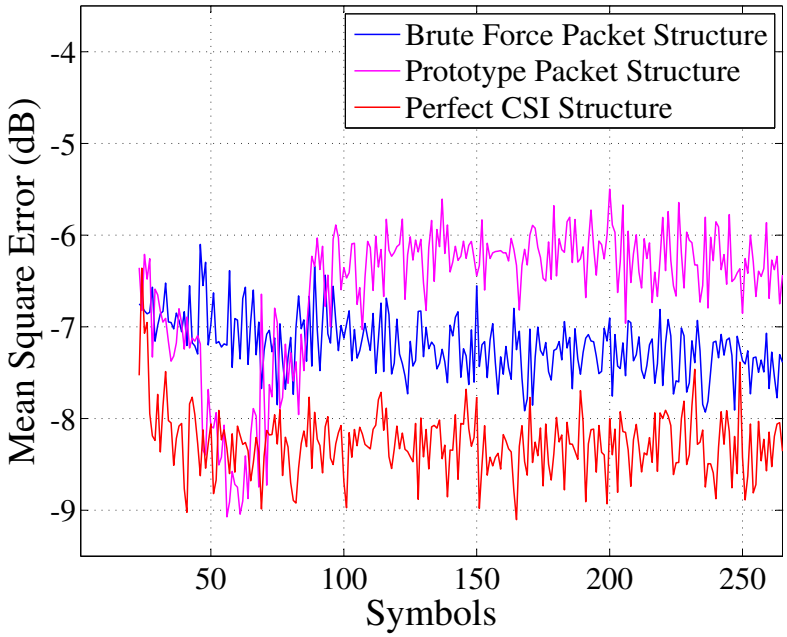

Figure 3.34 MOBILE CHANNEL: Packet structure comparison (10dB SNR)

at around 20 symbols and then climbs back to steady state at its starting MSE value around 100 symbols later. This is because the Prototype packet structure's training symbols are separated from the data symbols by a zero guard interval of length of v_h with zero energy (see Fig. 3.6). To guard against this null, we halt the adaptation of the Prototype structure's equalizer tap weights over the guard period.[3] Unfortunately, the "memory" of the equalizer still causes a small null in the MSE following the guard. During the guard period, the channel changes significantly such that the fully-trained tap weights no longer effectively equalize the decoded data, and we are back to where we started in terms of MSE.

The Brute Force packet structure of Fig. 3.4 has no guard symbols between its training symbols and data symbols. Therefore, its MSE trajectory appears smooth and continuous. Its poorer channel estimate training (L_{tce}) location at the beginning of the packet and shorter equalizer training sequence (L_{teq}) are more than made up

[3]Taking advantage of the linearity of the channel, a possible way to eliminate this zero energy interval would be to truncate the output observation sequence by adding the output samples corresponding to the L_g guard symbols to the succeeding output samples corresponding to the first L_g symbols of the data. The new superimposed samples would have the correct signal energy, but, unfortunately, twice as much noise energy.

for, compared to the Prototype structure, by having a longer L_{tce} (see Table 3.11). The norm of the difference between the perfect and estimated CSI for the Prototype structure averaged over 100 packets is 0.0033, which more than double the norm for the Brute Force structure of 0.0013.

We conclude that despite the hypothesized advantages of the Prototype packet structure, equal bandwidth efficiency-based simulation results have shown that the Brute Force structure, albeit less elegant, offers far superior MSE performance. Thus, we recommend using the conventional Brute Force method for the practical implementation of the TR-STBC system with a DFE.

3.5.5 Effect of Method of Steepest Descent Kick-Start on MSE Performance

Having seen the improvements in MSE convergence produced by reusing the tap-weights of the previous packet in Sec. 3.5.4, we conclude that this technique is effective for all but the the very first packet sent. However, for the first packet we can apply the Kick-Start Method of Steepest Descent proposed in Sec. 3.2.2 to boost the convergence speed and thus lower the MSE for the data portion of this packet. Our aim is to achieve the best trade-off between faster convergence rate and increased complexity by calibrating the recursive Method of SD algorithm's step-size to minimize the required number of iterations.

The Kick-Start Method of SD is applicable to both channel models, but we limit our simulation study to the MOBILE CHANNEL. We compare the MSEs of our Novel Hybrid DFEs running the CE-LMS algorithm with and without the Kick-Start Method of SD. The Monte Carlo simulations were carried out for the Perfect CSI packet structure commencing with two iterations of the Kick-Start Method of SD, where the SD step-size was set to $\mu = 0.008$.

In Fig. 3.35 we compare the MSE trajectories with and without the Kick-Start Method of SD. We see that two iterations of the Kick-Start Method of SD can lower the MSE by up to 2dB for the first 60 symbols. This means that the Kick-Start Method of SD would improve the decoding accuracy of the first packet significantly for a packet structure with a training sequence of less than 60 symbols. We postulate that it would

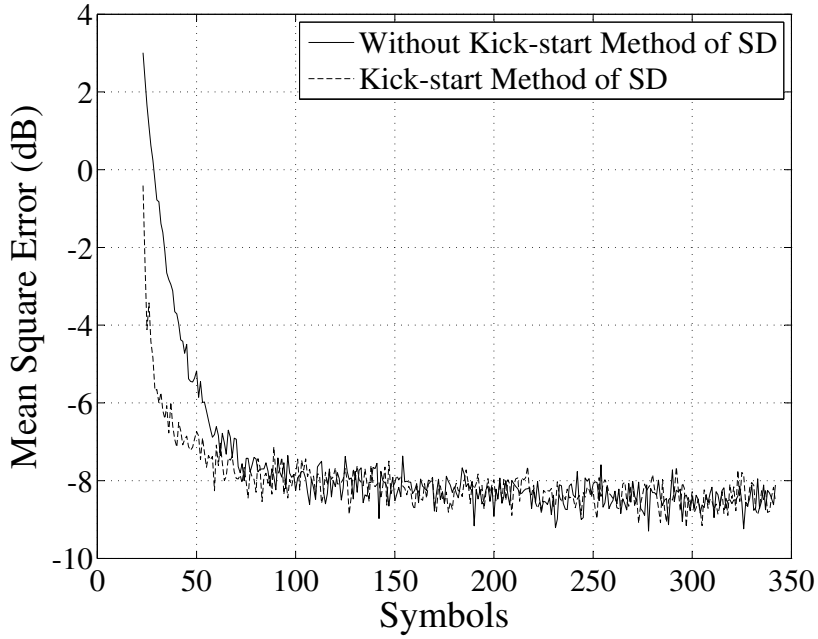

Figure 3.35 MOBILE CHANNEL: Kick-Start Method of Steepest Descent (10dB SNR)

be worthwhile to implement the Kick-Start Method of SD to enhance the initial training of the Brute Force packet structure for the MOBILE CHANNEL and the Prototype packet structure for the FIXED channel.

3.5.6 Dynamic Convergence Behavior of NLMS- and RLS-Based Algorithms

The simulated MSE performances of the SISO DFE architecture running the LMS algorithm and the Hybrid DFE structure running the CE-LMS algorithm were presented in Sec. 3.5.3. The results supported our theoretical analysis with the Hybrid DFE outperforming the SISO DFE. Therefore, we anticipate that the hybrid extensions of the Normalized LMS and Recursive Least Squares algorithms, the Combined Error-NLMS (CE-NLMS), Combined Estimate-RLS (CE-RLS) and Averaged Estimate-RLS (AE-RLS) algorithms, introduced in Sec. 3.3 should also outperform their conventional algorithm counterparts.

To compare the NLMS and RLS algorithms with their hybrid extensions, we simulate both the FIXED and MOBILE CHANNELS over 100 packets at 5dB and 30dB SNRs for the Perfect CSI and Prototype packet structures. We also include in our plots

Table 3.13 FIXED CHANNEL: Equalizer Simulation Parameters

Parameters	**5dB SNR**	**30dB SNR**
LMS Step-size	μ=0.0005	μ=0.001
NLMS Step-size	μ_o=0.6	μ_o=0.8
RLS λ	λ=0.01	λ=0.1
RLS δ	δ=0.99	δ=0.99

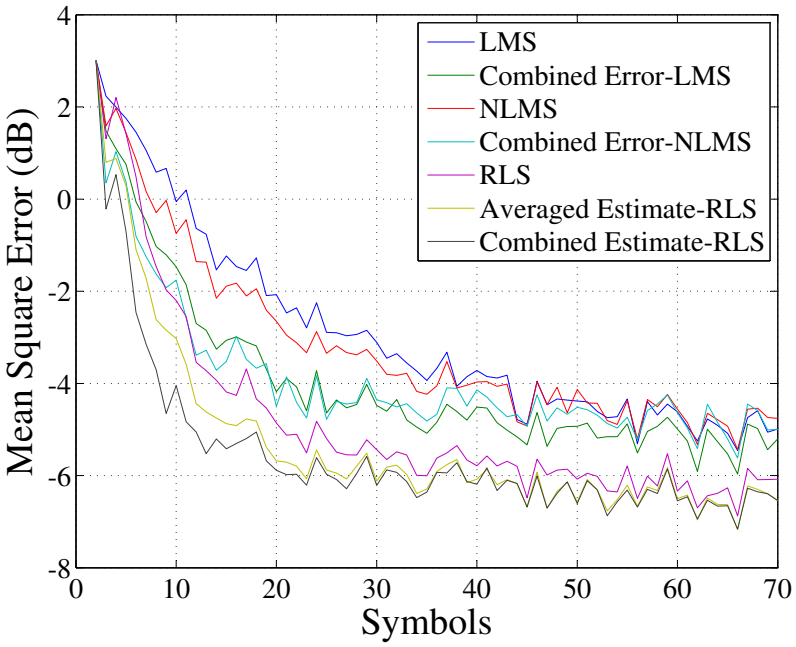

Figure 3.36 FIXED CHANNEL: Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (5dB SNR)

the simulated MSE trajectories of the LMS and CE-LMS algorithms from Sec. 3.5.3. This enables us to compare the MSE performances of all of the equalization algorithms presented in this thesis with each other.

For the FIXED CHANNEL, the tap weights are re-initialized to zero for every new packet, since the FIXED CHANNEL of Fig. 3.7 is not changing slowly over consecutive packets like the MOBILE CHANNEL of Fig. 3.9. The equalizer parameters for the LMS, NLMS and RLS algorithms and their hybrid counterparts are listed in Table 3.13. We have optimized the parameters individually for the lower and upper operating SNR bounds of 5dB SNR and 30dB SNR. For the Perfect CSI packet structure, the results are respectively shown in Figs. 3.36 – 3.37. At 5dB SNR, the hybrid algorithms in

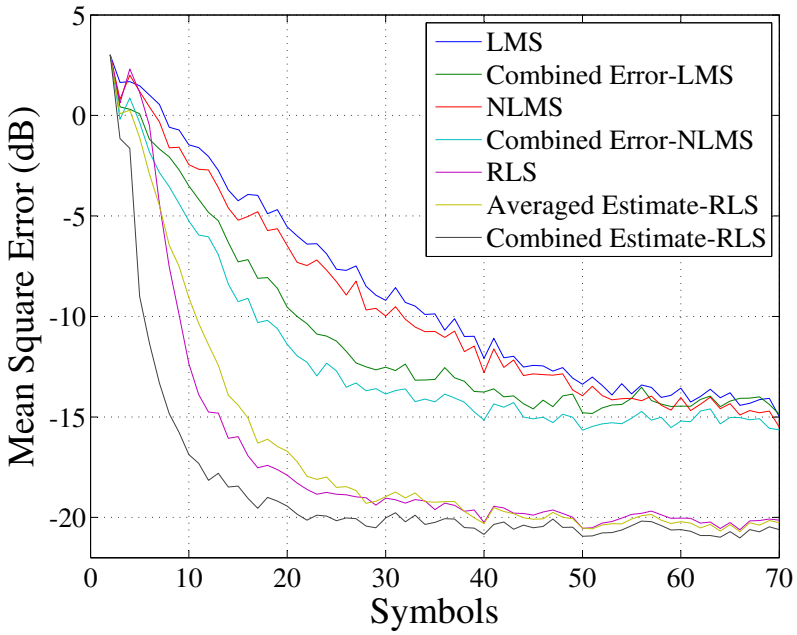

Figure 3.37 FIXED CHANNEL: Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (30dB SNR)

Fig. 3.36 display significantly faster convergence than their corresponding conventional algorithms. After the first 30 symbols, the CE-LMS and CE-NLMS algorithms hold 2dB MSE margins over the conventional LMS and NLMS algorithms, respectively. The CE-LMS algorithm also performs as well as the CE-NLMS algorithm at 5dB SNR, without the extra computational complexity of two dividers. After the first 10 symbols, the AE-RLS and CE-RLS algorithms hold 1 and 2dB MSE margins over the RLS algorithm, respectively. For the 30dB SNR in Fig. 3.37, the hybrid algorithms increase their convergence rate leads and achieve around 5dB lower MSEs than their conventional counterparts in the aforementioned packet locations. The CE-RLS algorithm is by far the best performing algorithm over the range of SNRs. However, it also ranks as one of the highest complexity algorithms.

We now study the equivalent convergence plots for the Prototype packet structure in Figs. 3.38 – 3.39. For the FIXED CHANNEL, the Prototype packet architecture results of Fig. 3.38 yield higher MSEs than the perfect CSI results. This falls in line with the previous packet structure comparisons shown in Fig. 3.34 for the MOBILE CHANNEL. At 5dB SNR the MSE trajectories for the various forms of the LMS and

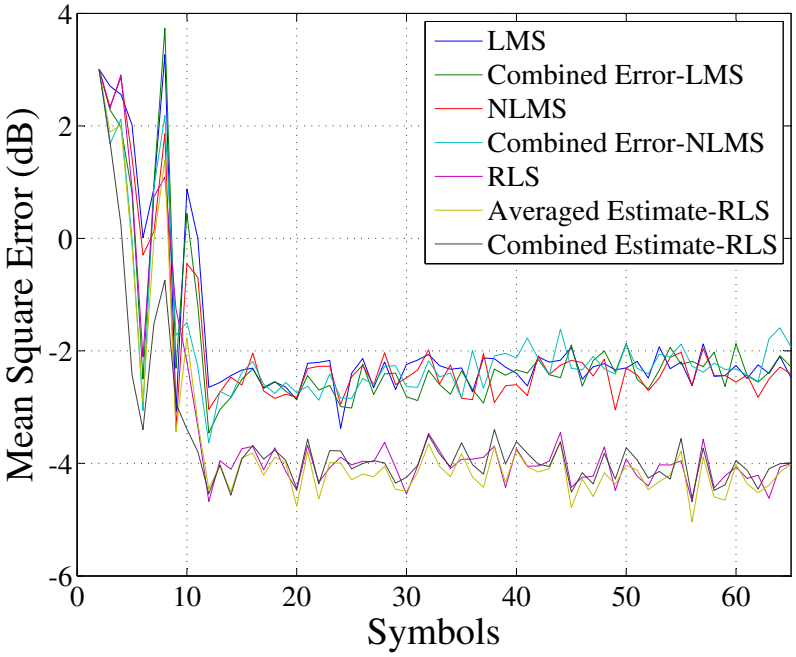

Figure 3.38 FIXED CHANNEL: Equalizer algorithm MSE convergence comparison for Prototype packet structure (5dB SNR)

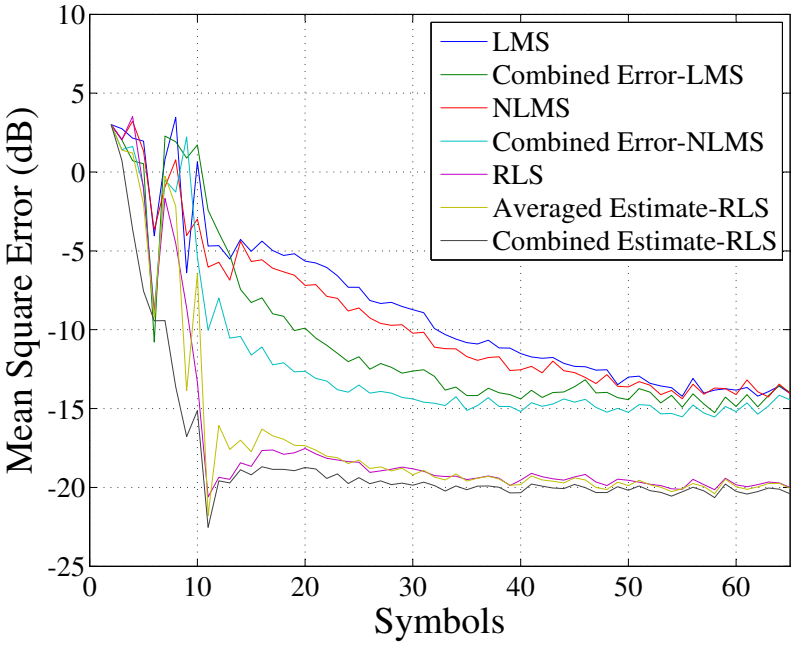

Figure 3.39 FIXED CHANNEL: Equalizer algorithm MSE convergence comparison for Prototype packet structure (30dB SNR)

Table 3.14 MOBILE CHANNEL: Equalizer Simulation Parameters

Parameters	**5dB SNR**	**30dB SNR**
LMS Step-Size μ	0.001	0.002
NLMS Step-Size μ_o	0.3	0.3
RLS λ	0.01	0.1
RLS δ	0.99	0.99

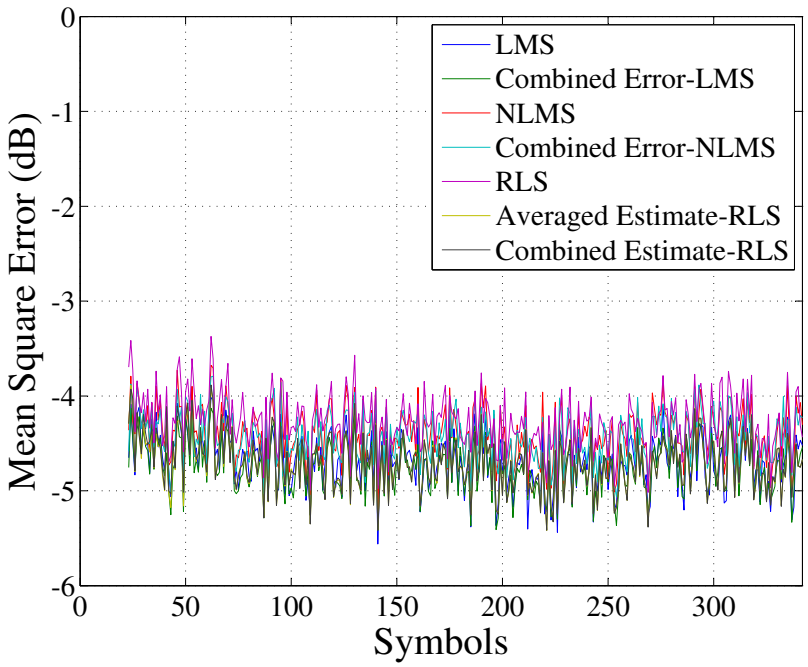

Figure 3.40 MOBILE CHANNEL: Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (5dB SNR)

NLMS algorithms all converge to -2.5dB, while the trajectories for the various RLS algorithms converge slightly lower at -4dB. At 30dB SNR Fig. 3.39 shows that the hybrid algorithms have reclaimed their 5dB steady-state MSE advantage over their conventional counterparts.

We now study the MSE performances of the hybrid versions of the NLMS and RLS algorithms for the MOBILE CHANNEL. For the first packet the tap weights are initialized to zero, whereas for subsequent packets the tap weights are initialized using the final tap weights for the previous packet. The algorithm parameters are optimized individually for 5dB SNR and 30dB SNR with their values listed in Table 3.14. For the Perfect CSI Structure the convergence results are shown in Figs. 3.40 – 3.41.

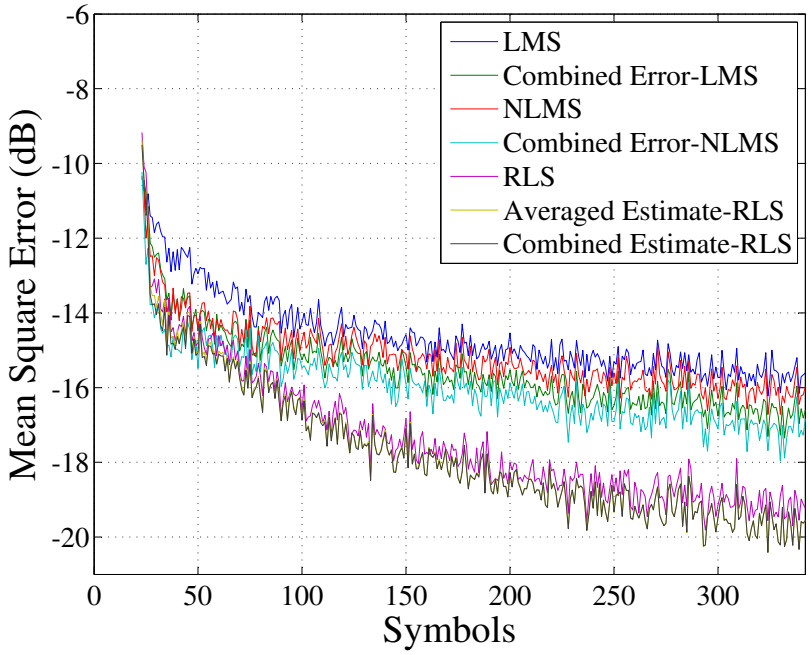

Figure 3.41 MOBILE CHANNEL: Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (30dB SNR)

At 5dB SNR in Fig. 3.40, there is not much difference among the various algorithms. At 30dB SNR in Fig. 3.41, the hybrid algorithms yield faster convergence than their corresponding conventional algorithms. However, the steady-state MSE margin is less than that for the FIXED CHANNEL. For the LMS-based algorithms the MSE margin is less than 2dB and for the RLS algorithms the margin is negligible. Similar to the FIXED CHANNEL, the highest complexity CE-RLS algorithm is again the best performing algorithm. The CE-LMS algorithm again provides very good convergence performance relative to the CE-NLMS algorithm and has the lowest computational complexity.

We display the Prototype packet structure MSE curves in Fig. 3.42 – 3.43. The nulls at the start of the packet in the figures are once again caused by the zero guard between the training and data sections of the packet. In this scenario we observe that the relative performances of the hybrid algorithms to their conventional counterparts are similar to their relative performances for the Perfect CSI Structure scenario. At 5dB SNR all of the MSEs line up at 2dB, which is 2dB higher than for the Perfect CSI case. The LMS-based hybrid algorithms gain a 1dB MSE margin at 30dB SNR, while

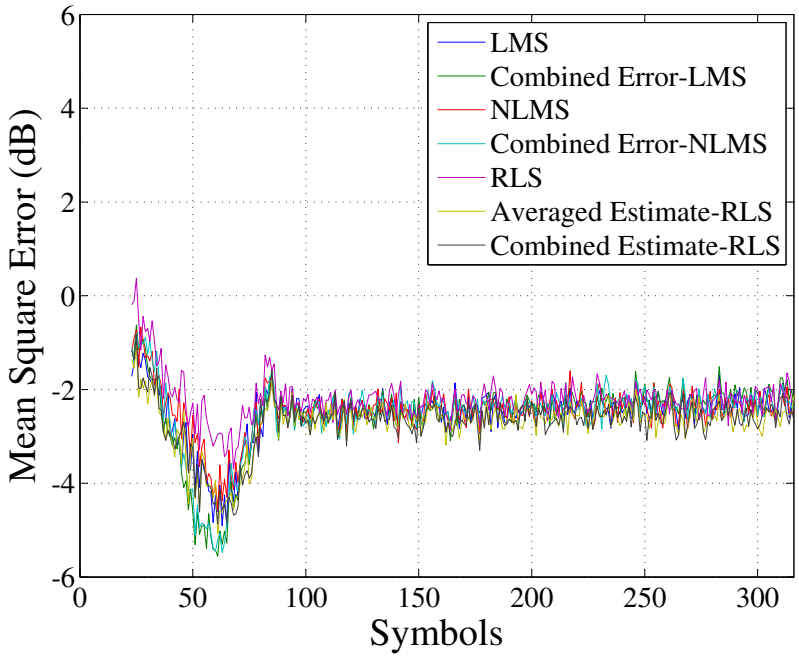

Figure 3.42 MOBILE CHANNEL: Equalizer algorithm MSE convergence comparison for Prototype packet structure (5dB SNR)

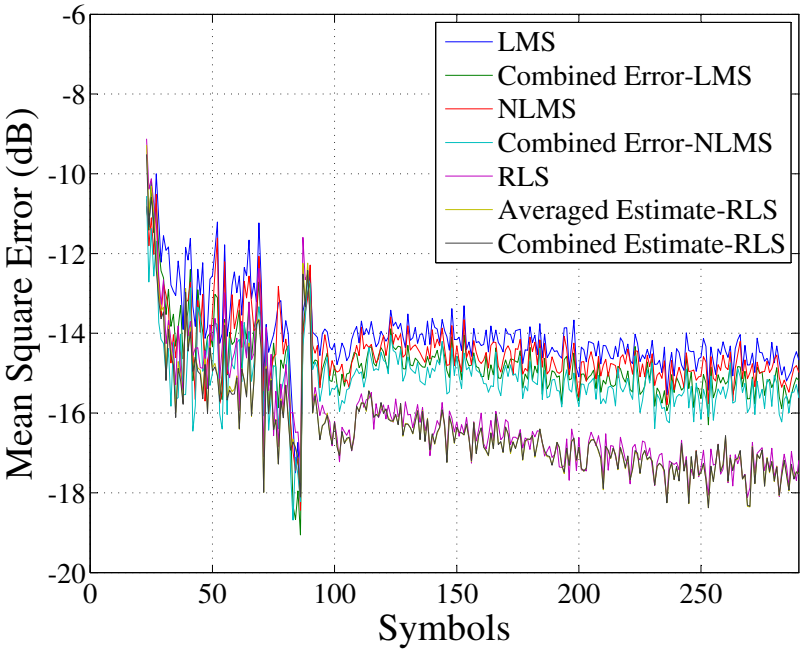

Figure 3.43 MOBILE CHANNEL: Equalizer algorithm MSE convergence comparison for Prototype packet structure (30dB SNR)

the RLS-based algorithms once again produce negligible advantage.

We conclude that the hybrid algorithms outperform their conventional algorithms for both the FIXED and MOBILE CHANNELS irrespective of the packet structure. The difference margin is greater for the FIXED CHANNEL than for the MOBILE channel and is greater for the LMS-based algorithms than for the RLS-based algorithms. The algorithm which best exploits the hybrid extension over all SNRs is the LMS algorithm, and it does it with the least complexity.

3.5.7 Steady-State Performances of DFEs in Mobile Channel

Thus far we have concentrated on the dynamic convergence behavior of the different DFE architectures and algorithms, using the instantaneous MSE at the equalizer output as our measure. It is also worth comparing the steady-state behaviors of the DFEs, using the Symbol Error Rate (SER) of the data payload as a function of input SNR as another measure. We now compare the steady-state performance of the Brute Force packet structure with the Prototype packet structure for the ITU Vehicular A Channel. In addition, we are particularly interested in the effect of Doppler for mobile radio channels, so we compute SER curves for four different Doppler values (max Doppler = 120km/hr), using the Prototype packet structure and ITU channel.

The Monte Carlo simulation setup for the steady-state simulations differs from the dynamic convergence simulation setup as follows:

1. Simulations are over an SNR range of 5–20dB in steps of 5dB.
2. Simulations are over 2000 packets for SNRs of 5–15dB and 45000 packets for 20dB.

These two adjustments are due to the rare occurrence of decision-device errors at high SNRs. We overcome this simulation difficulty by increasing the number of packets and lowering the highest SNR value. In these steady-state simulations, we again initialize the equalizer tap weights to zero for the first packet and to the final tap weights of the previous packet for subsequent packets.

The SER plot for a constant maximum vehicle speed of 120km/hr (555Hz Doppler) with the Prototype packet structure is shown in Fig. 3.44. We observe that there is

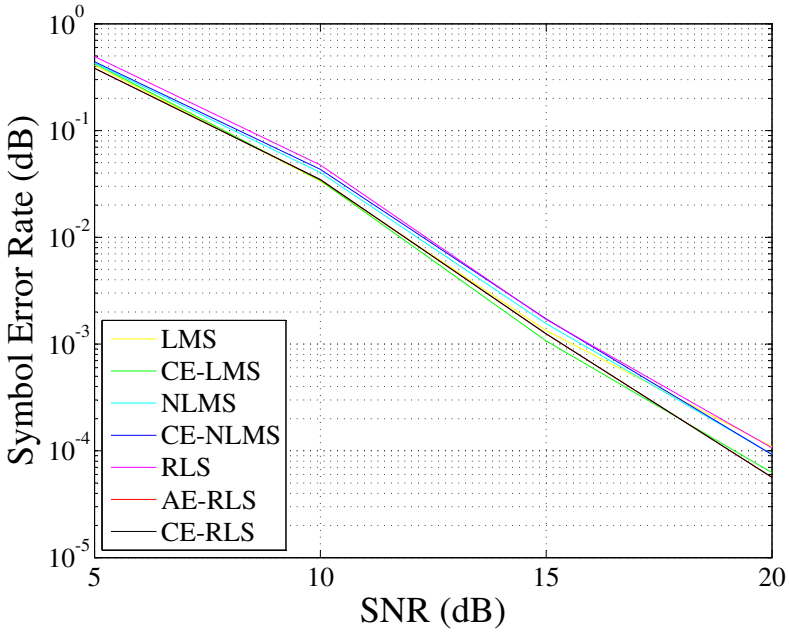

Figure 3.44 MOBILE CHANNEL: Steady-state performance of DFEs with 120km/hr (maximum) vehicular speed (Prototype packet)

very little steady-state difference between the various equalization algorithms. In fact, the CE-LMS algorithm seems to be performing the best for the least computational complexity. Therefore, we designate the Hybrid DFE running the CE-LMS algorithm as the benchmark equalizer when next studying the effects of Doppler.

We investigate the effect of Doppler or vehicle speed on the Prototype packet structure's steady-state performance by simulating the SER curves for four different Doppler values. The Monte Carlo simulation results for the Hybrid DFE for different maximum vehicle speeds are shown in Fig. 3.45. We observe that the SER curves follow two patterns. First, for each individual Doppler curve, the greater the SNR is, the lower the SER is. Second, the Doppler curves are stacked in order of speed: the lower the Doppler is, the lower the SER curve is. For an SER of 10^{-4}, at half the top speed (60km/hr) we obtain a 3dB SNR improvement, whereas at a quarter of the top speed (30km/hr) we obtain a 5dB SNR improvement.

For the Brute Force packet structure, the SER curves for the different DFE structures and algorithms with a maximum vehicle speed of 120km/hr (555Hz Doppler) are shown in Fig. 3.46. We observe from Fig. 3.46 that, similar to the Prototype packet

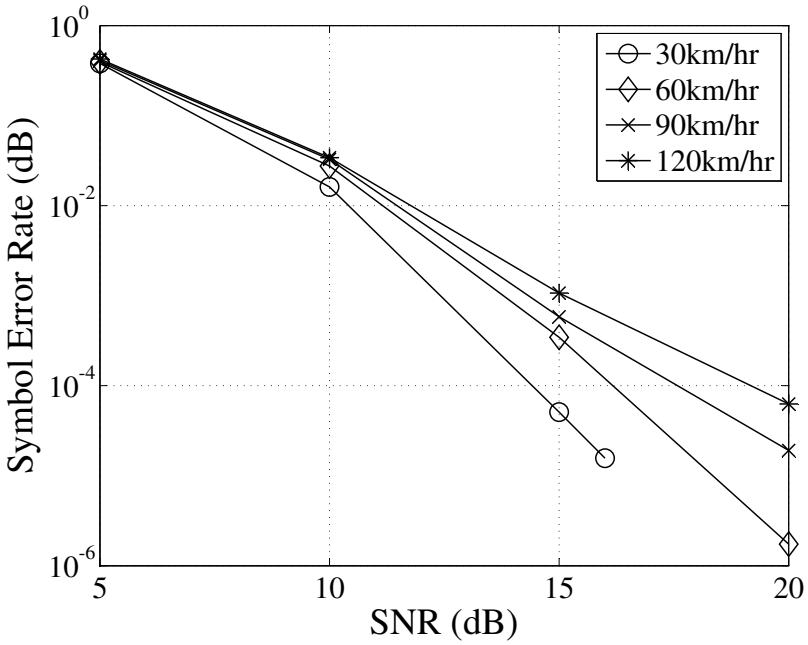

Figure 3.45 MOBILE CHANNEL: Doppler effect on steady-state performance of DFEs (Prototype packet)

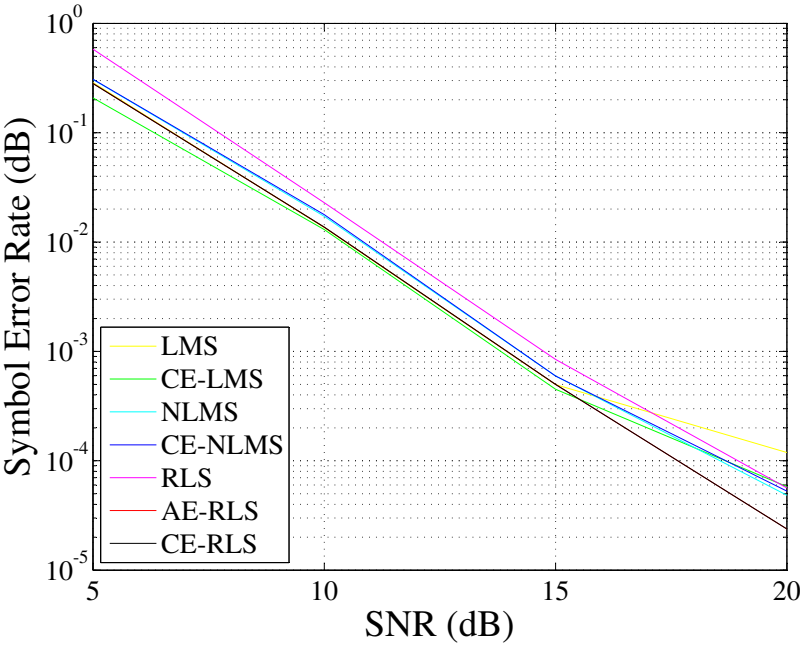

Figure 3.46 MOBILE CHANNEL: Steady-state performance of DFEs with 120km/hr (maximum) vehicular speed (Brute Force packet)

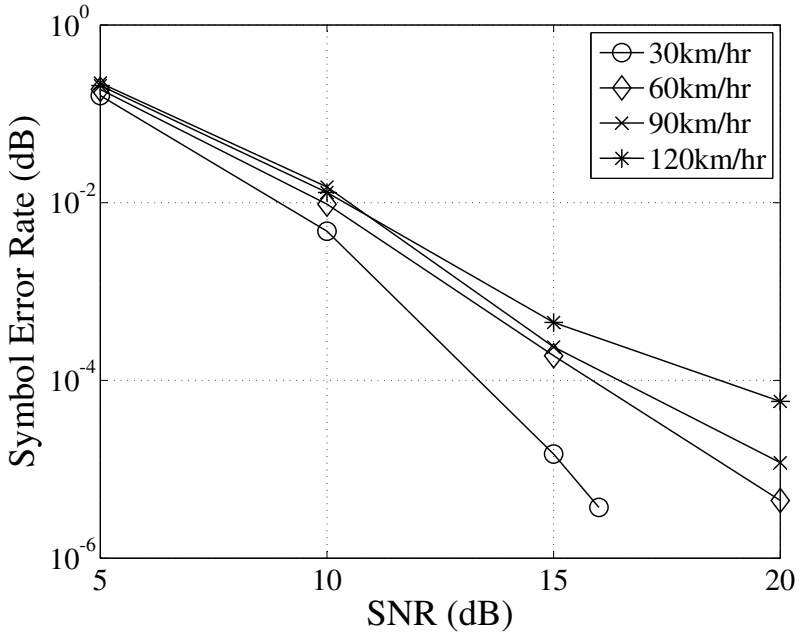

Figure 3.47 MOBILE CHANNEL: Doppler effect on steady-state performance of DFEs (Brute Force packet)

curves in Fig. 3.44, there is very little steady-state difference between the various equalization algorithms. Once again, the Hybrid DFE running the CE-LMS algorithm is performing the best for the least computational complexity. Therefore, we designate the Hybrid DFE as the benchmark equalizer when studying the effects of Doppler.

The effect of the maximum Doppler frequency on the Brute Force packet structure's steady-state performance is shown in Fig. 3.47. Again the SER curves in Fig. 3.47 show similar patterns as those in Fig. 3.45 for the Prototype packet structure. The relationship between the Doppler and the SER curves is less regular, but the curves are still stacked in order of speed: the lower the Doppler is, the lower the SER curve is. At SER $= 10^{-4}$, the lowest speed (30km/hr) yields a 5dB SNR gain over the highest speed (120km/hr). From the Doppler simulation results for both the Prototype and Brute Force packet structures we have verified the well-known property that the performance of a communication system over a mobile radio channel suffers as the Doppler spread increases.

Lastly, we compare the Prototype packet structure with the Brute Force packet at different maximum vehicular speeds in Fig. 3.48. For a given SER, irrespective of

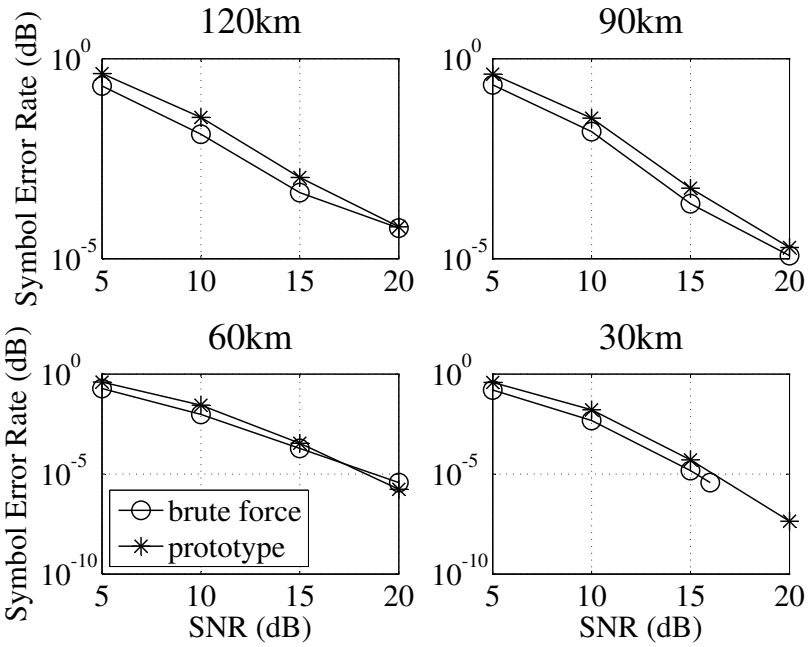

Figure 3.48 MOBILE CHANNEL: Comparison of Doppler effect on steady-state MSE for Prototype and Brute Force packet structures

the vehicle speed, the Brute Force packet structure's steady-state MSE performance in Fig. 3.48 is always approximately 2dB superior in SNR to the corresponding Prototype structure's steady-state MSE. This reconfirms the previous result of Fig. 3.34.

In this section we have characterized and optimized the MSE performances of the Parallel SISO, Hybrid SISO and MIMO DFEs running the LMS, NLMS and RLS algorithms and their corresponding hybrid extensions. We have shown for the MOBILE CHANNEL that, although the Prototype packet structure we proposed is easier to implement in hardware, its MSE performance is not as good as that of the Brute Force packet structure. We have also demonstrated the usefulness of the Kick-start Method of SD for the very first TR-STBC packet that is sent. Most importantly, the simulated MSE trajectories for both channel models have supported all our theoretical analysis and have shown that our new common error-based algorithms for the Hybrid DFE are superior to their conventional algorithm counterparts.

Chapter 4

DFE STRUCTURES AND ALGORITHMS FOR A $2 \times N_R$ TR-STBC SYSTEM

In Chapters 2 and 3 we have concentrated on the basic 2×1 TR-STBC system which was originally proposed by Alamouti [10] and extended by Lindskog and Paulraj [1]. However, our goal is to maximize the diversity gain or robustness of the wireless communication system against a fading channel. Thus far, we have only considered transmitter diversity techniques, but additional performance gains can be achieved using classical receiver diversity techniques as well. Receive diversity techniques have the advantage of not requiring additional transmit signal power, unlike transmit diversity techniques. Researchers such as Clark *et al.* [45] have shown that receiver diversity techniques can be successfully merged with MSE-minimizing equalization techniques. Therefore, we now extend the 2×1 TR-STBC equalization architectures and algorithms to a $2 \times N_R$ TR-STBC system. We first generalize our analysis and then use simulations to ascertain the benefits of additional receiver diversity.

4.1 ALTERNATIVE $2 \times N_R$ TR-STBC DFE STRUCTURES

At the receiver, instead of using a single antenna, we now use N_R antennas. Associated with each antenna is a TR-STBC decoder which forms two decoupled output streams, one containing observations due to $s_1(t)$ and the other containing observations due to $s_2(t)$. For example, the decoder for the i-th receive antenna produces streams $z_{1i}(t)$ and $z_{2i}(t)$. To yield a diversity gain while removing the residual ISI, the samples of $z_{1i}(t)$ and $z_{2i}(t)$, for $i = 1, \ldots, N_R$, must be fused together by the DFE. Here, we consider generalizations of the parallel SISO DFE and hybrid DFE structures proposed in Ch. 3.

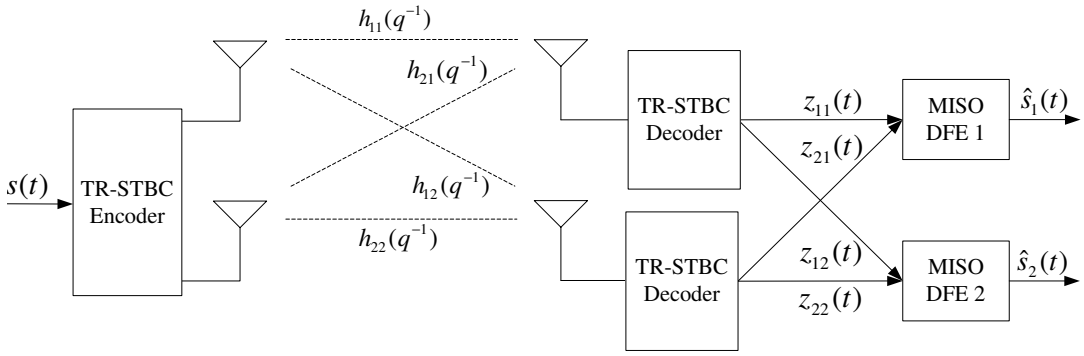

Figure 4.1 2×2 TR-STBC system with parallel MISO DFEs

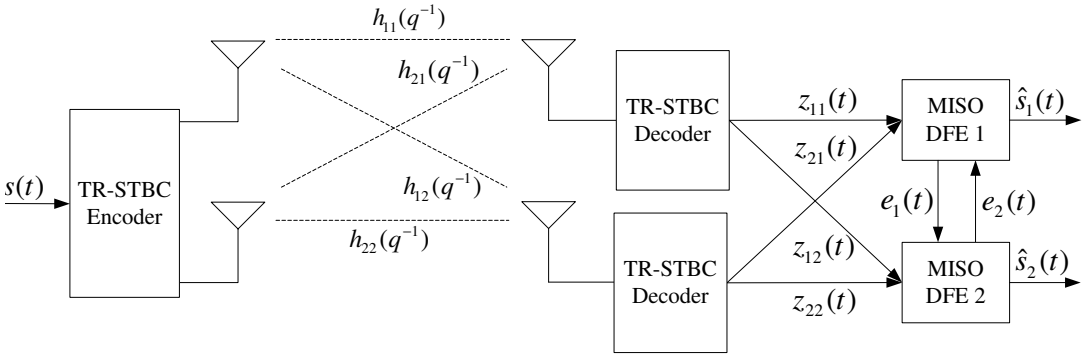

Figure 4.2 2×2 TR-STBC system with hybrid MISO DFE

As we saw no real advantage to using a MIMO DFE for the 2×1 system, we omit it in this chapter.

To combine the N_R output streams associated with each symbol stream, we now use two $N_R \times 1$ MISO equalizers in place of the two parallel SISO equalizers. For the special case of two receive antennas, the parallel MISO generalization of the parallel SISO DFEs is shown in Fig. 4.1. Similarly, the hybrid MISO generalization of the hybrid DFE is shown in Fig. 4.2.

4.1.1 MISO DFE Architecture

By replacing the single feedforward filter of the SISO DFE in Fig. 2.4 with N_R parallel feedforward filters, we can construct the MISO DFE structure shown in Fig. 4.3. We represent the fractionally-spaced tap weights for the i-th feedforward filter using length

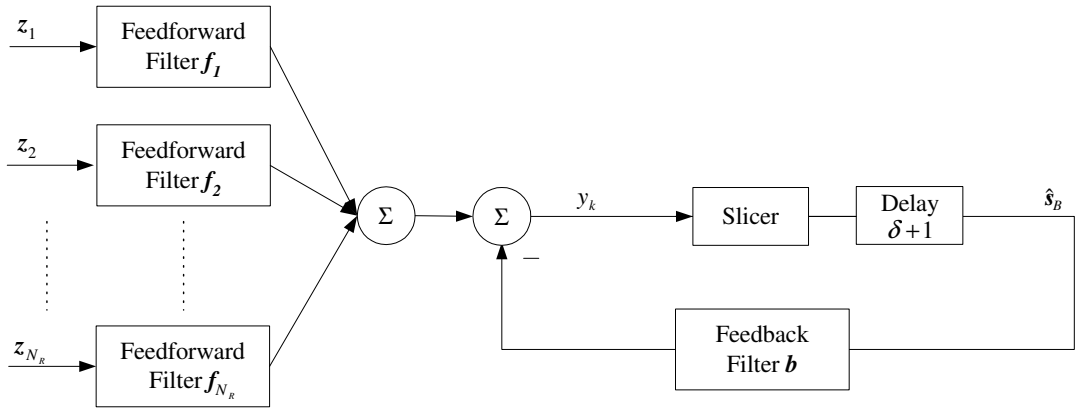

Figure 4.3 Block diagram of MISO DFE

$N_f P$ vector $\boldsymbol{f}_i$. Similarly, we represent the sample-spaced observations in the i-th filter using length $N_f P$ vector $\boldsymbol{z}_i$. Stacking all of the filter tap weights of the MISO DFE, both feedforward and feedback, into a single super vector in a similar manner to (2.53), we can write

$$\check{\boldsymbol{w}} = \begin{bmatrix} \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{N_R} \\ \boldsymbol{b} \end{bmatrix} \triangleq \begin{bmatrix} \check{\boldsymbol{f}} \\ \boldsymbol{b} \end{bmatrix}, \quad \check{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_{N_R} \\ -\boldsymbol{s}_B \end{bmatrix} \triangleq \begin{bmatrix} \check{\boldsymbol{z}} \\ -\boldsymbol{s}_B \end{bmatrix} \tag{4.1}$$

where $\boldsymbol{s}_B$ is defined as before. Replacing $\boldsymbol{w}$ and $\boldsymbol{v}$ by $\check{\boldsymbol{w}}$ and $\check{\boldsymbol{v}}$ in the notation of Sec. 2.3.6, we can easily derive the Wiener filter tap weights for this MISO DFE. We now consider the specific equalizer structures for the $2 \times N_R$ TR-STBC system.

4.1.2 Parallel MISO DFEs

In a similar manner to Sec. 3.1.1, we first analyze the two decoupled parallel MISO DFEs in Fig. 4.1. Given TR-STBC output processes $z_{1i}(t)$ and $z_{2i}(t)$, for $i = 1, \ldots, N_R$, we derive the Wiener filter tap weights for this architecture and then characterize the behavior of the LMS algorithm used to update the tap weights.

If we denote the channel impulse responses for signals traveling from the two transmit antennas to the i-th receive antenna as $h_{1i}(t)$ and $h_{2i}(t)$, we can define the effective

channel response at the output of the i-th TR-STBC decoder as

$$g_i(t) = \widetilde{h}^*_{1i}(t) * h_{1i}(t) + \widetilde{h}^*_{2i}(t) * h_{2i}(t)\,. \tag{4.2}$$

As in Sec. 3.1.1, by the Sampling Theorem, we can represent the continuous-time TR-STBC system using an equivalent discrete-time system with no loss of information. Again, we use matrix and vector notation to simplify the analysis. If we stack $N_f P$ oversampled observations of $z_{1i}(t)$ and $z_{2i}(t)$ into vectors $\boldsymbol{z}_{1i}$ and $\boldsymbol{z}_{2i}$, respectively, we can write

$$\boldsymbol{z}_{1i} = \boldsymbol{G}_i\,\boldsymbol{s}_1 + \widetilde{\boldsymbol{H}}_{1i}\,\boldsymbol{n}_{1i} + \boldsymbol{H}_{2i}\,\widetilde{\boldsymbol{n}}_{2i} \tag{4.3}$$

$$\boldsymbol{z}_{2i} = \boldsymbol{G}_i\,\boldsymbol{s}_2 + \widetilde{\boldsymbol{H}}_{2i}\,\boldsymbol{n}_{1i} - \boldsymbol{H}_{1i}\,\widetilde{\boldsymbol{n}}_{2i}\,. \tag{4.4}$$

Matrix $\boldsymbol{G}_i$ is of dimension $N_f P \times (N_f + v - 1)$ with

$$\boldsymbol{G}_i = \begin{bmatrix} \boldsymbol{g}_{1i} & \boldsymbol{g}_{2i} & \cdots & \boldsymbol{g}_{vi} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{g}_{1i} & \boldsymbol{g}_{2i} & \cdots & \boldsymbol{g}_{vi} & \boldsymbol{0} & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{g}_{1i} & \boldsymbol{g}_{2i} & \cdots & \boldsymbol{g}_{vi} \end{bmatrix} \tag{4.5}$$

where length P vector

$$\boldsymbol{g}_{ki} = [g_i[(k-1)P + P]\ g_i[(k-1)P + P - 1]\ \cdots\ g_i[(k-1)P + 1]]^T\ . \tag{4.6}$$

Matrix $\boldsymbol{H}_{ji}$ is the $N_f P \times (N_f P + v_h P - 1)$ standard channel convolution matrix corresponding to channel $h_{ji}(t)$, $j = 1, 2$, with Toeplitz form

$$\boldsymbol{H}_{ji} = \begin{bmatrix} h_{ji}(1) & h_{ji}(2) & \ldots & h_{ji}(v_h P) & 0 & \cdots & 0 \\ 0 & h_{ji}(1) & h_{ji}(2) & \cdots & h_{ji}(v_h P) & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{ji}(1) & h_{ji}(2) & \cdots & h_{ji}(v_h P) \end{bmatrix}\,. \tag{4.7}$$

Matrix $\widetilde{\boldsymbol{H}}_{ji}$ is formed by replacing $h_{ji}(t)$ by $\widetilde{h}^*_{ji}(t)$ in (4.7).

With the TR-STBC outputs in matrix format, we now derive the Wiener solution for the TR-STBC MISO DFEs. To derive the Wiener solution for MISO DFE 1, we substitute $\boldsymbol{z}_{1i}$, for $i = 1, \ldots, N_R$, of (4.3) into super vector $\check{\boldsymbol{v}}$ of (4.1) and form the DFE autocorrelation matrix. If we define $\check{\boldsymbol{z}}_1 = \left[\boldsymbol{z}_{11}^T \ \cdots \ \boldsymbol{z}_{1N_R}^T\right]^T$, the autocorrelation matrix has the form

$$\boldsymbol{R}_{1,\delta} = \begin{bmatrix} \mathrm{E}\{\check{\boldsymbol{z}}_1 \ \check{\boldsymbol{z}}_1^H\} & -\mathrm{E}\{\check{\boldsymbol{z}}_1 \ \boldsymbol{s}_B^H\} \\ -\mathrm{E}\{\boldsymbol{s}_B \ \check{\boldsymbol{z}}_1^H\} & \mathrm{E}\{\boldsymbol{s}_B \ \boldsymbol{s}_B^H\} \end{bmatrix}. \tag{4.8}$$

Assuming the symbol and noise processes are mutually independent and i.i.d., we can show that submatrix $\mathrm{E}\{\check{\boldsymbol{z}}_1 \ \check{\boldsymbol{z}}_1^H\}$ has the form

$$\begin{aligned} \mathrm{E}\{\check{\boldsymbol{z}}_1 \ \check{\boldsymbol{z}}_1^H\} &= \begin{bmatrix} \mathrm{E}\{\boldsymbol{z}_{11} \ \boldsymbol{z}_{11}^H\} & \mathrm{E}\{\boldsymbol{z}_{11} \ \boldsymbol{z}_{12}^H\} & \cdots & \mathrm{E}\{\boldsymbol{z}_{11} \ \boldsymbol{z}_{1N_R}^H\} \\ \mathrm{E}\{\boldsymbol{z}_{12} \ \boldsymbol{z}_{11}^H\} & \mathrm{E}\{\boldsymbol{z}_{12} \ \boldsymbol{z}_{12}^H\} & \cdots & \mathrm{E}\{\boldsymbol{z}_{12} \ \boldsymbol{z}_{1N_R}^H\} \\ \vdots & & \ddots & \vdots \\ \mathrm{E}\{\boldsymbol{z}_{1N_R} \ \boldsymbol{z}_{11}^H\} & \cdots & \cdots & \mathrm{E}\{\boldsymbol{z}_{1N_R} \ \boldsymbol{z}_{1N_R}^H\} \end{bmatrix} \\ &= \begin{bmatrix} \sigma_{s_1}^2 \boldsymbol{G}_1 \boldsymbol{G}_1^H + \beta_1 & \sigma_{s_1}^2 \boldsymbol{G}_1 \boldsymbol{G}_2^H & \cdots & \sigma_{s_1}^2 \boldsymbol{G}_1 \boldsymbol{G}_{N_R}^H \\ \sigma_{s_1}^2 \boldsymbol{G}_2 \boldsymbol{G}_1^H & \sigma_{s_1}^2 \boldsymbol{G}_2 \boldsymbol{G}_2^H + \beta_2 & \cdots & \vdots \\ \vdots & & \ddots & \sigma_{s_1}^2 \boldsymbol{G}_{N_R-1} \boldsymbol{G}_{N_R}^H \\ \sigma_{s_1}^2 \boldsymbol{G}_{N_R} \boldsymbol{G}_1^H & \cdots & \sigma_{s_1}^2 \boldsymbol{G}_{N_R} \boldsymbol{G}_{N_R-1}^H & \sigma_{s_1}^2 \boldsymbol{G}_{N_R} \boldsymbol{G}_{N_R}^H + \beta_{N_R} \end{bmatrix} \end{aligned} \tag{4.9}$$

where

$$\beta_i = \sigma_{n_{1i}}^2 \widetilde{\boldsymbol{H}}_{1i} \widetilde{\boldsymbol{H}}_{1i}^H + \sigma_{\tilde{n}_{2i}}^2 \boldsymbol{H}_{2i} \boldsymbol{H}_{2i}^H. \tag{4.10}$$

The off-diagonal submatrix $\mathrm{E}\{\check{\boldsymbol{z}}_1 \ \boldsymbol{s}_B^H\} = \left[\mathrm{E}\{\boldsymbol{s}_B \ \check{\boldsymbol{z}}_1^H\}\right]^H$ can be written

$$\mathrm{E}\{\check{\boldsymbol{z}}_1 \ \boldsymbol{s}_B^H\} = \begin{bmatrix} \mathrm{E}\{\boldsymbol{z}_{11} \ \boldsymbol{s}_B^H\} \\ \vdots \\ \mathrm{E}\{\boldsymbol{z}_{1N_R} \ \boldsymbol{s}_B^H\} \end{bmatrix} = \sigma_{s_1}^2 \begin{bmatrix} \boldsymbol{G}_1(:, \delta+1 : \delta+N_b) \\ \vdots \\ \boldsymbol{G}_{N_R}(:, \delta+1 : \delta+N_b) \end{bmatrix} \tag{4.11}$$

and again matrix $\mathrm{E}\{\boldsymbol{s}_B \ \boldsymbol{s}_B^H\} = \sigma_{s_1}^2 \boldsymbol{I}_{N_b}$. Similarly, using the independence of the symbols

and noise, the cross-correlation vector $\boldsymbol{p}_{1,\delta}$ has the form

$$\boldsymbol{p}_{1,\delta} = \begin{bmatrix} \sigma_{s_1}^2 \boldsymbol{G}_1(:,\delta) \\ \vdots \\ \sigma_{s_1}^2 \boldsymbol{G}_{N_R}(:,\delta) \\ \boldsymbol{0}_{N_b \times 1} \end{bmatrix}. \tag{4.12}$$

To derive $\boldsymbol{R}_{2,\delta}$ and $\boldsymbol{p}_{2,\delta}$, similar expressions can be constructed. But, again using the assumptions $\sigma_{s_1}^2 = \sigma_{s_2}^2$ and $\sigma_{n_{1i}}^2 = \sigma_{n_{2i}}^2$, for $i = 1, \ldots, N_R$, and the property

$$\widetilde{\boldsymbol{H}}_{ji}\, \widetilde{\boldsymbol{H}}_{ji}^H = \boldsymbol{H}_{ji}\, \boldsymbol{H}_{ji}^H, \qquad j = 1, 2, \quad i = 1, \ldots, N_R \tag{4.13}$$

we can easily show that $\boldsymbol{R}_{2,\delta} = \boldsymbol{R}_{1,\delta}$ and $\boldsymbol{p}_{2,\delta} = \boldsymbol{p}_{1,\delta}$, yielding identical Wiener filter tap weights for the two MISO equalizers. With our new MISO versions of $\boldsymbol{R}_{1,\delta}$ and $\boldsymbol{p}_{1,\delta}$ shown in (4.8) and (4.12), we can finally form the Wiener filter tap weights (3.13) and the sum MMSE (3.16).

If we independently adapt each of the MISO DFEs using the conventional LMS algorithm (2.32), the MSE dynamic convergence behavior of the two equalizers can be approximated using the Independence Method trajectory (2.66) or the Small Step-size Method trajectory (2.77) with $\boldsymbol{R}_{i,\delta}$ and $\boldsymbol{p}_{i,\delta}$ calculated using (4.8) and (4.12), respectively. Finally, the sum MSE trajectory of the parallel MISO equalizers can be expressed using (3.17) with $\boldsymbol{R}_1$ replaced by (4.8).

4.1.3 Hybrid MISO DFE

As we have shown in Sec. 4.1.2, the Wiener filter solutions for the two MISO DFEs for the $2 \times N_R$ TR-STBC are identical. Thus, we can apply our hybrid combined-error approach to the MISO DFEs. Using the new super vectors in (4.1), we can directly implement the CE-LMS algorithm of (3.20) to update the common MISO filter tap weights. We derive the MSE dynamic convergence behavior for the $2 \times N_R$ system by substituting $\boldsymbol{R}_{1,\delta} = \boldsymbol{R}_{2,\delta}$ of (4.8) and $\boldsymbol{p}_{1,\delta} = \boldsymbol{p}_{2,\delta}$ of (4.12) into (3.22) – (3.26). For example, using the Small Step-size Method, the MSE trajectory for the $2 \times N_R$ system has the same form as that for the 2×1 system in (3.27).

To increase the dynamic convergence rate of the hybrid MISO equalizer, for the $2 \times N_R$ system we also investigate the combined error approach applied to the NLMS and RLS algorithms for the MISO DFE. Using our new super vector notation of (4.1), we can directly extend the SISO algorithms introduced in Sec. 3.3 to the hybrid MISO equalizer.

4.2 EQUALIZER COMPLEXITY ANALYSIS

Extending the TR-STBC system to N_R receive antennas increases the overall receiver complexity approximately N_R-fold. Because we require one channel estimator and decoder for each receive antenna, the decoder hardware complexity increases proportionally to N_R. Despite the fact that the number of MISO equalizers stays constant at two, both equalizers' feedforward section lengths, represented by $\check{\boldsymbol{f}}$ in the super tap weight vector (4.1), are proportional to N_R. Therefore, the most hardware consuming part of the equalizer still grows proportionally to the number of receivers.

Similar to Sec. 3.4, in this section we once again measure the complexity of our equalizer algorithms and architectures in terms of Multiply-ACcumulate (MAC) operations. In our hypothetical complexity study of the 2×1 TR-STBC system, all of the MIMO processing is handled by the EP1S25 FPGA apart from the adaptive filtering which is implemented using an auxiliary TMS320C6714 DSP. For the $2 \times N_R$ system we propose to upgrade to a larger and faster FPGA, with the minimum upgrade being the Altera EP1S80 FPGA. Operating at 120MHz the EP1S80 FPGA is theoretically able to deliver (in terms of MACs):

- 22 DSP blocks: 10560MMACs
- 176 9-bit multipliers: 10560MMACs
- 79040 uncommitted logic elements: 24000MMACs

which totals to over 45 billion MACs per second. This is three times greater than the EP1S25's computational capabilities, making the EP1S80 FPGA suitable for the implementation of, for example, a 2×2 TR-STBC system. We do not consider a hardware upgrade necessary for the adaptive filtering section of a 2×2 system.

The first step in TR-STBC decoding is to estimate the CIRs based on the observations due to the channel estimation training sequence. This was outlined in Sec. 3.4. Here we only emphasis that this process now needs to be carried out in parallel at N_R receivers for $2N_R$ channels. For the Kick-start Method of Steepest Descent, according to (4.8) – (4.12), we now have to compute expanded $\boldsymbol{R}_\delta$ matrices and $\boldsymbol{p}_\delta$ vectors for the two MISO DFEs, which roughly scale up by factors of N_R^2 and N_R in complexity, respectively.

4.2.1 Equalizer Complexity Comparison

We introduced two complexity components for our adaptive equalizers in Sec. 3.4: the complexity of computing the equalizer output for a given equalizer architecture and the complexity of recursively updating the equalizer tap weights using a particular adaptive filtering algorithm. The complexity of computing the output of a MISO DFE is shown in Table 3.5 with n replaced by N_R. This same complexity holds for both the parallel MISO and hybrid MISO DFE architectures.

To determine the complexity of recursively updating the equalizer tap weights, note that the computational complexity of the 2×1 versions of the adaptive algorithms are summarized in Table. 3.8, where $L = N_f P + N_b$. We can extend the table to represent the $2 \times N_R$ system simply by using the new substitution $L = N_R N_f P + N_b$. The complexities for the LMS-based algorithms increase linearly and proportional to N_R, whereas the complexities of the RLS-based algorithms increase quadratically due to their L^2 components.

4.3 2×2 TR-STBC SYSTEM SIMULATION RESULTS

In this section we describe receiver diversity equalization simulation results for a 2×2 TR-STBC system. In general, receiver diversity techniques suffer from diminishing returns. Therefore, extending the 2×1 TR-STBC system to a 2×2 TR-STBC system is most likely to yield the most performance enhancement for the least increase in complexity. We formulate our baseband simulations for the 2×2 TR-STBC system using the generalized $2 \times N_R$ DFE derivations in the Sec. 4.1, setting $N_R = 2$.

This section is organized similar to Sec. 3.5, omitting the analysis of the training enhancements with different packet structures and the Kick-start Method of SD. First, we redefine the wireless channel models and the equalizer parameters for the 2×2 TR-STBC system in Sec. 4.3.1. We begin our simulation analysis in Sec. 4.3.2, comparing our theoretical analysis to the simulated results for the three DFE architectures, as well as their dynamic convergence behaviors. We then study the dynamic convergence behaviors of the NLMS- and RLS-based algorithms in Sec. 4.3.3. Finally, we analyze and compare the steady-state performances of the Hybrid DFEs for the 2×1 and 2×2 systems in Sec. 4.3.4.

4.3.1 Wireless Channel Models and System/Equalizer Parameters

We extend the same channel models for the 2×1 TR-STBC system, described in Sec. 3.5.1, to the 2×2 TR-STBC system. Thus far we have assumed that there is no correlation between the transmit antennas. Here, we also assume that the receive antennas are spaced far enough so that there is no spatial correlation between the receivers. Therefore, we can simulate the four TR-STBC channels independently using the same channel parameters. For the FIXED CHANNEL we include two more captured 2×1 TR-STBC channels for the second receiver. For the MOBILE channel, we generate another pair of channels via the modified version of Smith's algorithm [42], with the tapped-delay line impulse response parameters for the ITU Vehicular A channel given by Table 3.9. Finally, to facilitate comparison, we keep the system and equalizer parameters the same as those for the 2×1 TR-STBC system outlined in Sec. 3.5.2.

4.3.2 LMS Dynamic Convergence Behavior for Three DFE Architectures

The main purpose of our simulations in this section is to show that the 2×2 transceiver delivers better performances than those for the 2×1 transceiver, and that our statistical analysis of the LMS algorithm for the various DFE architectures still applies for the $N_R = 2$ case. To isolate the effects of the different DFE structures on the dynamic convergence behavior of the equalizers, we perform all of our Monte Carlo simulations

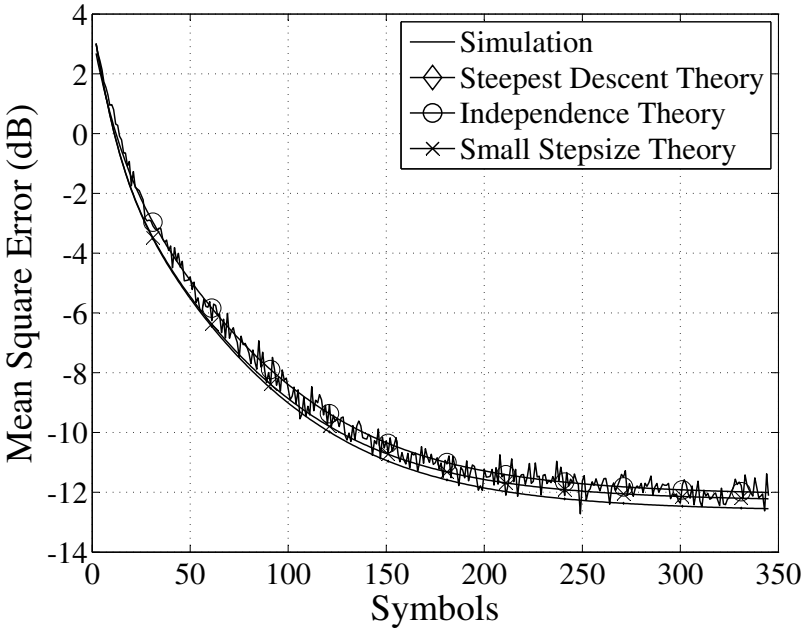

Figure 4.4 FIXED CHANNEL: MSE trajectory for Parallel MISO DFEs (10dB SNR)

for the case of perfect knowledge of the CSI by the TR-STBC decoder, averaging over 100 packets. We choose a 10dB SNR which enables us to compare the MSE convergence performances against those for the 2×1 system at the same SNR. We first use the FIXED CHANNEL model and then use the MOBILE CHANNEL model.

For the FIXED CHANNEL, the equalizer taps are initialized to zero for each new packet. The LMS step-size is $\mu = 0.00025$ for both MISO DFE structures. Considering first the Parallel MISO DFE architecture, we compare the simulated MSE trajectory with the statistical MSE trajectories for a 10dB SNR in Fig. 4.4. We observe that the Parallel MISO DFE MSE trajectories in Fig. 4.4 exhibit characteristics similar to the analogous SISO DFE trajectories in Fig. 3.10, except that the steady-state MSE is 2dB lower. The statistical MSE trajectories align closely with the simulated MSE trajectory, with the best-fit statistical line generated using Independence Theory. The Independence Theory trajectory consistently has the highest MSE, while the Steepest Descent trajectory is overly optimistic, ignoring the noisy estimates of the gradient.

We then consider the Novel Hybrid MISO DFE, comparing the simulated MSE trajectory with the analytical MSE trajectories for the FIXED CHANNEL in Fig. 4.5.

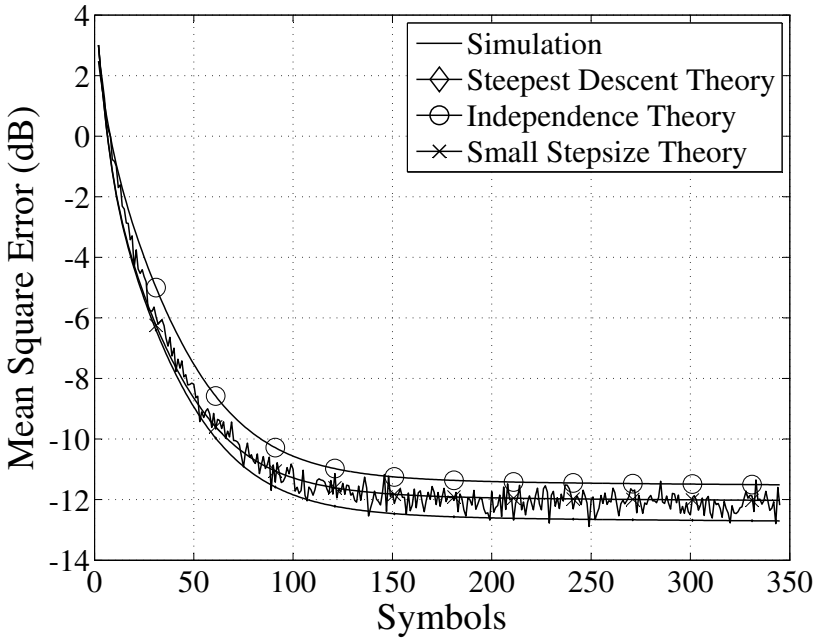

Figure 4.5 FIXED CHANNEL: MSE trajectory for Novel Hybrid MISO DFE (10dB SNR)

Once again the Hybrid MISO DFE trajectories in Fig. 4.5 bear strong resemblance to their analogous Hybrid SISO DFE curves in Fig. 3.13, except that the steady-state MSE is 2dB lower. The Hybrid MISO DFE's simulated MSE trajectory has closest alignment with the Small Step-size MSE trajectory, situated between the Independence Method and the Steepest Descent MSE trajectories.

We now proceed to the MOBILE CHANNEL, setting the step-size to $\mu = 0.001$ for 10dB SNR. We show the simulated and analytical MSE trajectories for the Parallel MISO DFEs in Fig. 4.6. We observe from Fig. 4.6 that the Independence Method most closely approximates the simulated MSE, sitting 0.5dB below the simulated trajectory. We also note that the simulated MSE trajectory for the Parallel MISO DFEs is 2dB lower than the corresponding trajectory for the Parallel SISO DFEs in Fig. 3.19.

For the Novel Hybrid MISO DFE we plot the simulated and theoretical MSE trajectories in Fig. 4.7. We see that for the Hybrid DFE the simulated MSE trajectory once again aligns most closely with the Independence Method trajectory. Again, the MSE trajectories for the Novel Hybrid MISO DFE are 2dB lower than those for the Novel Hybrid SISO DFE in Fig. 4.7.

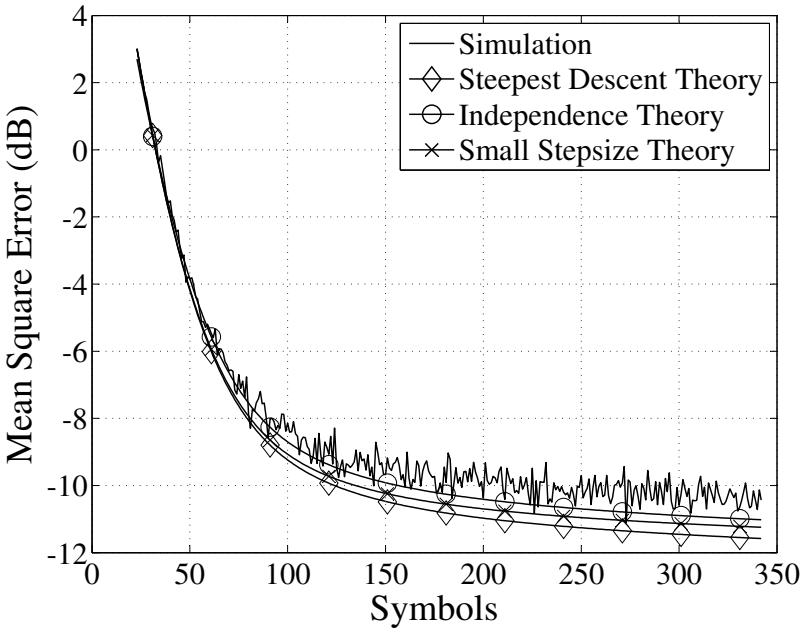

Figure 4.6 MOBILE CHANNEL: MSE trajectory for Parallel MISO DFEs (10dB SNR)

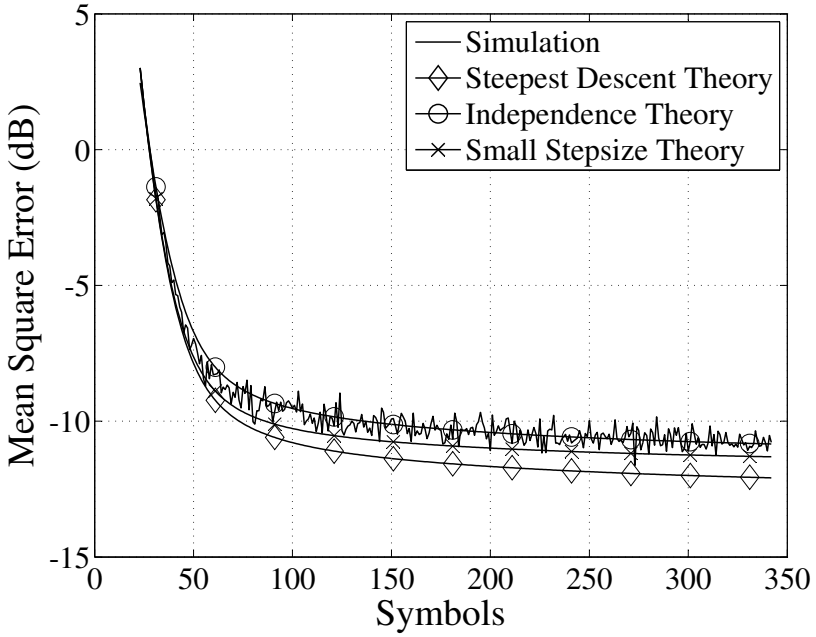

Figure 4.7 MOBILE CHANNEL: MSE trajectory for Novel Hybrid MISO DFE (10dB SNR)

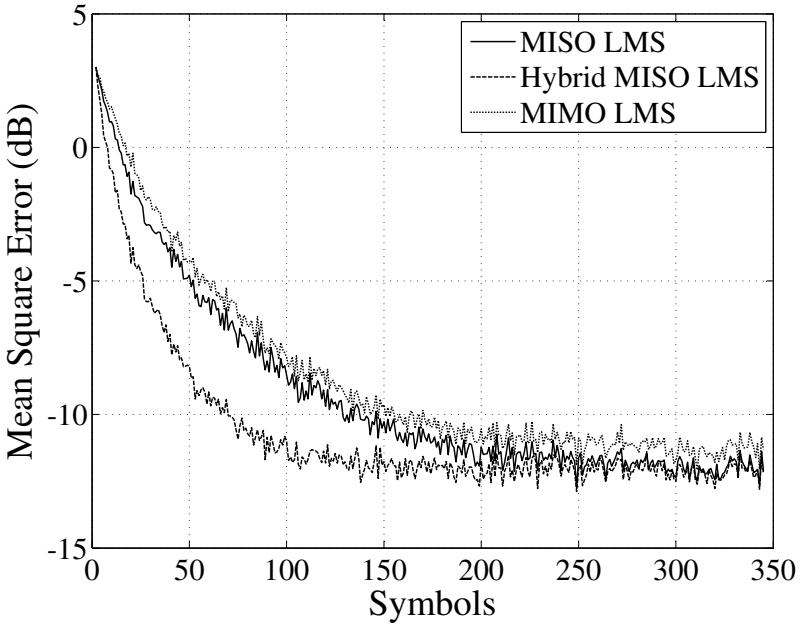

Figure 4.8 FIXED CHANNEL: MISO DFE architecture comparison (10dB SNR)

Finally, we compare the simulated MSE trajectories of the two MISO DFE architectures at 10dB SNR for the FIXED and MOBILE CHANNELS respectively in Figs. 4.8 and 4.9. We bring back the MIMO DFE just to keep a consistent comparison with the corresponding plots for the 2×1 system at 10dB SNR. For both the FIXED and MOBILE CHANNELS the Novel Hybrid MISO DFE produces the fastest convergence to steady-state, yielding an MSE margin of around 3dB over both the Parallel MISO DFEs and the MIMO DFE after the first 100 symbols. This margin is 1dB greater than the analogous Hybrid DFE margin for the 2×1 system. The respective MSE steady-state values for the three equalizers are also 2dB lower than those in the 2×1 plots of Figs. 3.31 – 3.31.

In summary, we have demonstrated, using Monte Carlo simulations of the dynamic LMS convergence behaviors for both the FIXED and MOBILE CHANNELS, that the two alternative statistical convergence theories yield better MSE convergence approximations to the simulated MSE trajectory than the original Steepest Descent Theory for the 2×2 system. Because our analysis holds for any general N_R, we can assume this approximation would also extend to the MISO DFEs for any $2 \times N_R$ TR-STBC system. We conclude that for any $2 \times N_R$ TR-STBC system the Novel Hybrid MISO

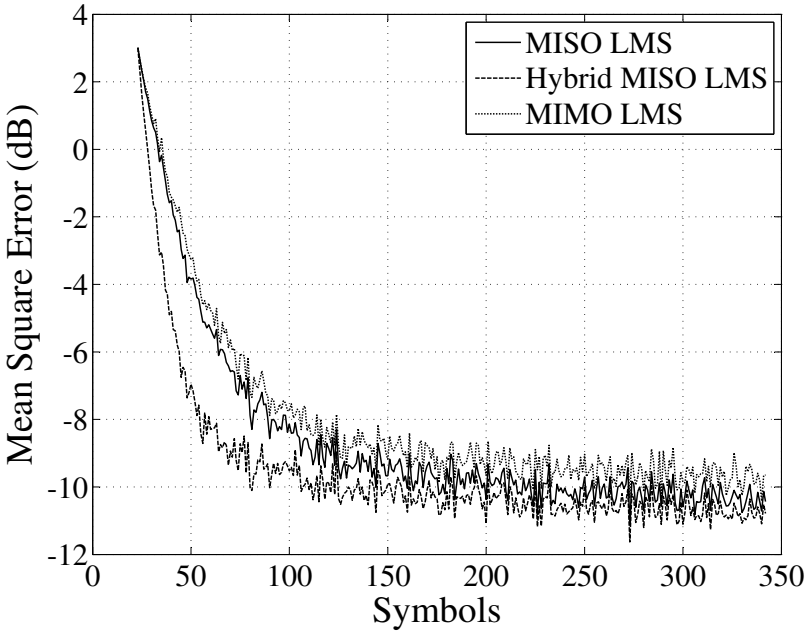

Figure 4.9 MOBILE CHANNEL: MISO DFE architecture comparison (10dB SNR)

DFE yields the best equalizer performance for a given complexity.

4.3.3 Dynamic Convergence Behavior of NLMS- and RLS-Based Algorithms

The MSE performances of the Parallel MISO DFE architecture running the LMS algorithm and the Hybrid MISO DFE structure running the CE-LMS algorithm were presented in Sec. 4.3.2. The results support our theoretical analysis with the Hybrid MISO DFE's performance superseding that of the Parallel MISO DFE. Therefore, we anticipate that the CE-NLMS, CE-RLS and AE-RLS algorithms introduced in Sec. 3.3 should outperform their conventional algorithm counterparts.

Here we simulate both the FIXED and MOBILE CHANNELS over 100 packets at 5dB and 30dB SNRs. We evaluate the NLMS and RLS algorithms and their hybrid extensions for the Perfect CSI packet structure. Similar to the 2 × 1 system simulations, we also include in our plots the simulated trajectories for the LMS and CE-LMS algorithms from Sec. 4.3.2 to enable us to compare the MSE performances of all of the equalization algorithms.

For the FIXED CHANNEL the tap weights are re-initialized to zero for every new packet. The equalizer parameters for the LMS, NLMS and RLS algorithms and their

Table 4.1 FIXED CHANNEL: 2 × 2 Equalizer Simulation Parameters

Parameters	**5dB SNR**	**30dB SNR**
LMS Step-size	μ=0.00025	μ=0.0005
NLMS Step-size	μ_o=0.4	μ_o=0.8
RLS λ	λ=0.01	λ=0.1
RLS δ	δ=0.99	δ=0.99

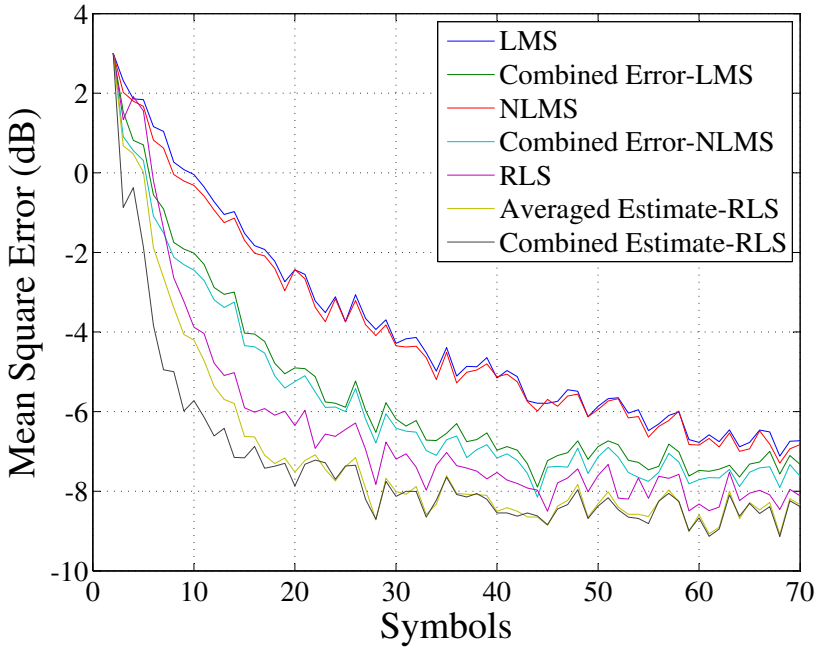

Figure 4.10 FIXED CHANNEL: 2 × 2 Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (5dB SNR)

hybrid counterparts are listed in Table 4.1. The parameters are optimized individually for SNRs of 5 and 30dB. For the Perfect CSI packet structure, the results are respectively shown in Figs. 4.10 and 4.11. At 5dB SNR, the hybrid algorithms in Fig. 4.10 converge significantly faster than their conventional counterparts. The CE-LMS and CE-NLMS algorithms hold a MSE margin of 2dB for much of the packet length of 75 symbols over respectively the LMS and NLMS algorithms, compared to for only 20 symbols for the 2×1 plot of Fig. 3.36. Once again, the CE-LMS algorithm achieves the same the MSE performance as the CE-NLMS algorithm at 5dB SNR, without the extra computational complexity of two dividers. The AE-RLS algorithm's MSE trajectory again sits between those for the RLS and CE-RLS algorithms. The CE-RLS algorithm

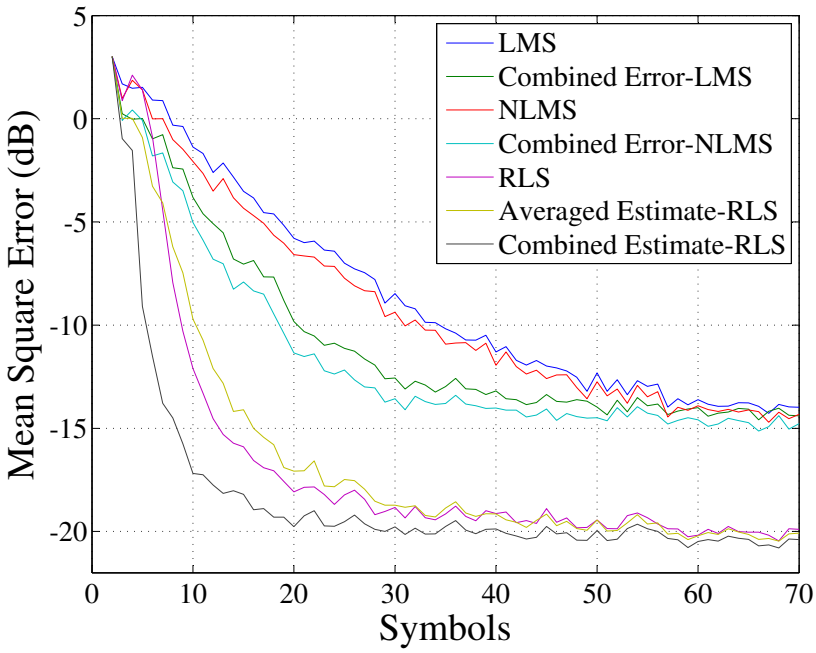

Figure 4.11 FIXED CHANNEL: 2 × 2 Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (30dB SNR)

holds a 2dB MSE margin over the RLS algorithm after the first 20 symbols. For 30dB SNR in Fig. 3.37, the hybrid algorithms increase their lead in convergence rate and are over 5dBs lower, similar to their analogous trajectories in the 2 × 1 plot.

At the low SNR of 5dB, we also observe that, overall, the MSE trajectories and steady-state values for the 2 × 2 system are 2dB lower than those for the 2 × 1 system. However, at the high SNR of 30dB, there is negligible advantage gained by the additional receive diversity. The CE-RLS algorithm is still by far the best performing algorithm over the range of SNRs, as well as being one of the highest complexity algorithms.

We now study the NLMS and RLS Hybrid algorithms' MSE performances for the MOBILE CHANNEL. The tap weights are initialized to zero for the very first packet and to the final tap weights of the previous packet for subsequent packets. The algorithm parameters are identical to those for the 2 × 1 system shown in Table 3.14. For the Perfect CSI packet structure, the convergence results are shown in Figs. 4.12 and 4.13. At 5dB SNR in Fig. 4.12, there is little difference between the various algorithms, with the exception of the RLS algorithm's MSE being 0.5dB above the rest

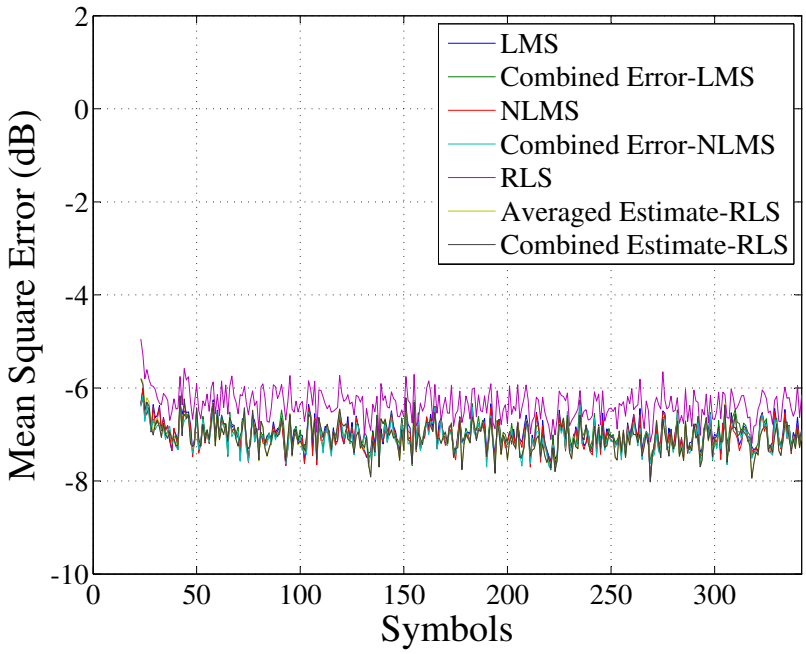

Figure 4.12 MOBILE CHANNEL: 2 × 2 Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (5dB SNR)

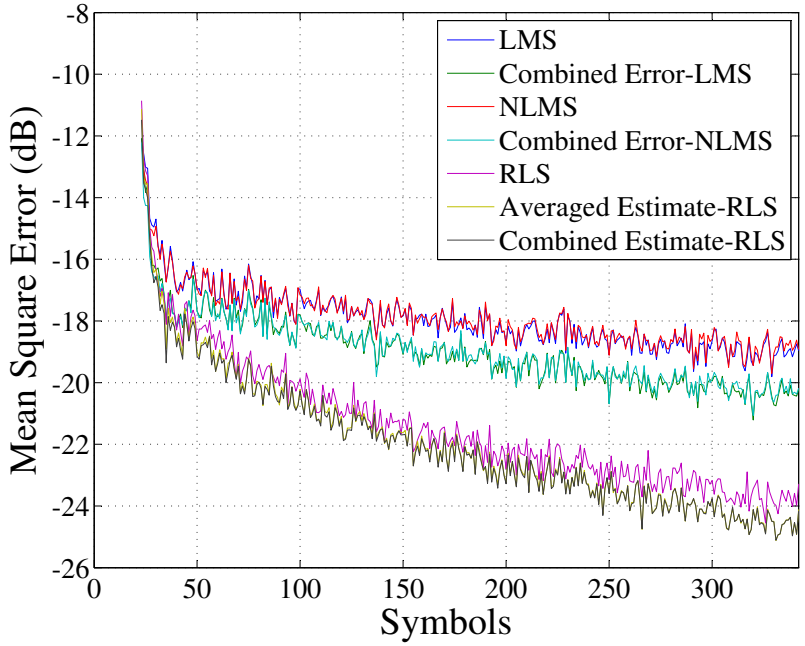

Figure 4.13 MOBILE CHANNEL: 2 × 2 Equalizer algorithm MSE convergence comparison for Perfect CSI packet structure (30dB SNR)

of the MSEs. At 30dB SNR in Fig. 4.13, the hybrid algorithms converge faster than their corresponding conventional algorithms. However, the margin is less than that for the FIXED CHANNEL. For the LMS-based algorithms the MSE margin is 1dB and for the RLS algorithms the margin is 0.5dB. Similar to the FIXED CHANNEL, the highest complexity CE-RLS algorithm is again the best performing algorithm. The low complexity CE-LMS algorithm again provides very good convergence performance relative to the CE-NLMS algorithm.

We conclude that the hybrid algorithms outperform the conventional algorithms for both the FIXED and MOBILE CHANNELS for the 2×2 TR-STBC system. This result we can also theoretically extend to the $2 \times N_R$ case. The difference margin is once again greater for the FIXED CHANNEL than the MOBILE channel and greater for the LMS-based algorithms than the RLS-based algorithms. The algorithm which best exploits the hybrid extension over all SNRs is again the low complexity LMS algorithm.

4.3.4 Comparison of Steady-State Performances of DFEs in Mobile Channel for $N_R \geq 1$

Thus far, we have shown that, by exploiting the receiver diversity gain, the DFE architectures and algorithms for the 2×2 system achieve dynamic convergence performance gains of roughly 2dB over their 2×1 system counterparts. Again, we are also interested in the relative steady-state performances of the two systems. As before, we use the Symbol Error Rate (SER) performance measure to compare the two systems.

We simulate the benchmark Hybrid MISO DFE ($\mu = 0.001$) for the 2×2 TR-STBC system with the Prototype packet structure in the ITU Vehicular A Channel with a maximum Doppler of 120km/hr. Our results are displayed in Fig. 4.14. We see that at SER=10^{-1}, the steady-state SNR of the 2×2 system is 1.5dB less than that of the 2×1 system. This difference grows to 3dB for SER=10^{-2}, 4.5dB for SER=10^{-3} and approximately 7dB for SER=10^{-4}. Since every 3dB represents a doubling of SNR, to achieve an SER of 10^{-4}, the 2×2 system only requires one quarter of the transmitted signal power required by the 2×1 system. The cost is that any $2 \times N_R$ system is N_R times more computationally complex than the 2×1 system. Fortunately, here we have

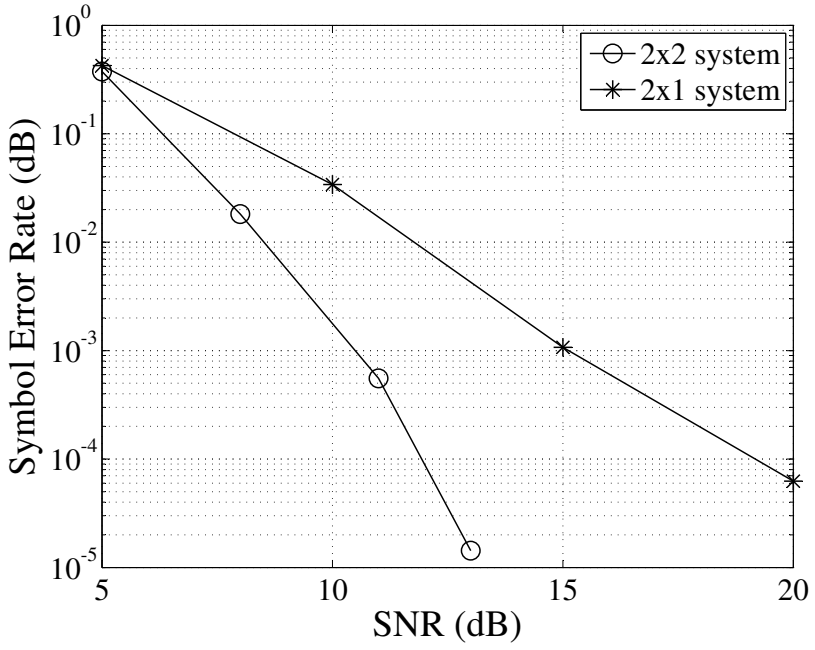

Figure 4.14 MOBILE CHANNEL: Steady-state comparison of DFEs for 2×1 vs. 2×2 TR-STBC systems with 120km/hr (maximum) vehicular speed (Prototype packet)

found a nonlinear gain in steady-state performance for a linear increase in complexity.

We conclude that a 2×2 system is a very attractive option in terms of base station-to-mobile antenna real estate and has an excellent performance versus complexity trade-off. For $N_R \geq 2$ we would most likely observe diminishing returns for the extra hardware and complexity.

Chapter 5

CONCLUSION

5.1 RESEARCH SUMMARY

Practical implementation of STBC systems is currently an area of great interest for wireless equipment manufacturers. The reliability of STBC systems can be increased by using diversity techniques combined with an equalizer at the receiver to eliminate the inter-symbol interference caused by multipath propagation. In this thesis, we have devised and studied practical equalizer structures for TR-STBC-based schemes, which are considered promising candidates for indoor transmission as well as for the EDGE system.

Exploiting the common second-order statistics of the TR-STBC decoder output streams, we have proposed a new Combined Error approach to equalizer design, which has yielded computationally efficient architectures and algorithms that converge faster than their conventional counterparts. Using Monte Carlo simulations we have investigated both the dynamic and steady-state performances of our alternative DFEs for both a captured indoor FIXED CHANNEL and an ITU standard MOBILE CHANNEL.

A good understanding of the dynamic converge behavior of an equalizer is crucial for choosing the proper lengths of the channel estimation and equalizer training sequences within a packet. There is a tradeoff between having long enough sequences for good equalization performance and keeping them short enough to preserve the bandwidth efficiency of the communication system. To satisfy both sides of the tradeoff, ideally an equalizer should train up quickly to a good steady-state performance. Using statistical analysis and computer simulations, we have found that our new Combined Error algorithms do this using a reasonable computational complexity.

Specific dynamic behavior results that we have obtained in this thesis are as follows. We have found that the Hybrid SISO DFE running the CE-LMS algorithm converges faster to a better steady-state MSE than both the conventional Parallel SISO DFEs and the MIMO DFE. Imposing a TR-STBC structure on the channel estimation training sequence, we have proposed a new Prototype packet structure. Using simulations to compare this structure with the conventional Brute Force structure, we have concluded that the accuracy of the channel estimates for use in the TR-STBC decoder is crucial for good overall performance. Although the Prototype structure can lead to simpler hardware implementations, we have found that it is better to use the Brute Force structure due to its longer channel estimation sequences within each packet.

As a technique to speed up the initial convergence of the equalizer, we have proposed the Kick-Start Method of Steepest Descent. Using simulations we have shown that this method is a useful way to jump-start the equalizer during the very first packet. However, for subsequent packets, due to the continuous behavior of the time-varying channel, we have discovered that it is better to use the final equalizer tap weights of the previous block to initialize the tap weights.

To speed up the equalizer convergence rate even more, we have proposed new Combined Error versions of the NLMS and RLS algorithms. We have found that these new algorithms do increase the convergence rate, but at the cost of increased complexity over LMS-based algorithms. If we factor in the complexity, we have shown that the CE-LMS algorithm is quite competitive, as it has the greatest MSE performance improvement over its conventional counterpart. Also, it performs as well as the CE-NLMS algorithm without requiring the latter's divide operations.

System engineers are also interested in the steady-state performance of the system experienced by the data payload. Therefore, we run SER simulations for our equalizers. We have found that the steady-state performances are relatively independent of the adaptive algorithms and DFE architectures, suggesting that the simpler LMS-based algorithms are adequate from a steady-state vantage point. However, we suspect that the steady-state performances are more dependent on static parameters such as the equalizer length and DFE structure. This would be worth investigating further. Finally,

studying our equalizers for different maximum Doppler frequency values, we have found the usual SER performance degradation as the Doppler increases.

In the last part of the thesis, we have considered the effects of adding conventional receiver diversity to the transmitter diversity system. We have derived hybrid MISO equalizer architectures and algorithms which show similar dynamic convergence behaviors to their hybrid SISO counterparts. Using simulations of a 2×2 system, we have shown that significant diversity gains can be achieved over a 2×1 system at the cost of twice the computational complexity. With our focus on practical implementation, we conclude that the 2×2 system with the Hybrid MISO DFE is the the most ideal in terms of performance gain versus complexity.

5.2 FUTURE WORK

This thesis has focused on the structural and theoretical optimization of TR-STBC systems for practical implementation. We have only briefly covered the implementation aspects of the system. Further analysis is needed to simplify the DFE architectures for practical implementation. The next step would be to simulate and optimize the system in fixed point, followed by implementation in hardware on the proposed FPGA platforms. It would be interesting to make performance versus complexity comparisons with the Tait STAR platform using the optimal Viterbi equalizer [11].

We can also apply the Combined Error approach to equalizers for other communication systems with similar coupled but identical decoder output signal statistics. One candidate is the Space-Frequency Block-Coded (SFBC) Single Carrier-Frequency Domain Equalization (SC-FDE) system of [46]. Like a STBC system, the SFBC system uses two antennas, but this time it encodes two data blocks over two adjacent frequency carriers instead of over two consecutive symbol intervals. Similar to the DFE for the TR-STBC system, the corresponding equalizer for the SFBC system is also based on a MMSE error criterion, except that it operates in the frequency domain.

Younis *et al.* [47] have recently proposed a block method of combining the time reversal-space time block decoding and the equalization into a single operation. The adaptive algorithm for their FDE concurrently trains the time reversal-based block

decoder as well as the equalizer. In a practical system requiring both training and data sequences, due to their block structure, it is very difficult to combine both training symbols and data symbols in the same packet. Thus, unlike for our symbol-based approach, for their scheme packets have to be devoted either to training or to data. This is potentially a great disadvantage for fast time-varying channels where a close combination of the training and data symbols is crucial. By transferring their concurrent algorithm to the time domain, more flexible packet structures can be developed which would work well for fast time-varying channels. Applying our Combined Error approach in this context would be a worthwhile enterprise.

Finally, a Single Carrier Frequency Domain Equalization-STBC(SC FDE-STBC), utilizing a linear MMSE criterion based DFE [17] is a viable competitor that should be compared with our equalizer in the future. Instead of having a known training sequence, the FDE-STBC uses a preamble that is equal the last L symbols of the transmitted block. The complexity of the signal processing associated with FDE is low, as it is mostly done via FFT. One disadvantage of an FDE is that the equalizer length is fixed to the data block length, regardless of the channel impulse response. The FDE's DFE will be much more complex than our DFE for short CIRs, since our equalizer can be quite short for these cases.

Appendix A

TR-STBC TRAINING-BASED CHANNEL ESTIMATION

A TR-STBC transceiver encodes a sequence of data symbols which is then transmitted by two antennas and received by one antenna. The receiver requires an accurate estimate of the channel impulse response (CIR) to be able to decode the STBC-encoded data. To estimate the CIR, a training sequence is sent as part of each data packet. An estimate is then formed based on the channel observations corresponding to the training sequence.

In this appendix, we discuss the derivation of the best training sequence for CIR estimation. We first review the classical least-squares approach to training sequence design and then outline a new technique proposed by Fragouli *et al.* in [39]. This technique imposes the same TR-STBC structure on the training sequence that is imposed on the data portion of each transmitted block. Although the resulting training sequence is suboptimal compared to the unconstrained case, it achieves a performance close to that of the optimal unconstrained training sequence.

There are several advantages to this scheme. First, the best QPSK training sequence can be found using an exhaustive search within a reasonable amount of time. Second, the structure of the training sequence can be implemented using the same TR-STBC encoding that the data uses, reducing the transmitter complexity. Finally, this TR-STBC structure for the training sequence enables it to be used to train the DFE after the TR-STBC decoder as well.

A.1 BACKGROUND

To derive the best training sequence, we modify our channel model notation to accommodate processing prior to the TR-STBC decoder. Again, we assume a quasi-static CIR that is constant over each transmitted data packet. We let two training sequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, both containing L_t symbols, to be simultaneously transmitted over the two channels $h_1(k)$ and $h_2(k)$. If the receiver samples the channel output using a fractional spacing of P samples per symbol and the CIR has a duration of v_h symbols, we can stack the v_hP samples of the first CIR $h_1(k),\ k = 0, \ldots, v_hP-1$ in the following matrix

$$\boldsymbol{H}_1 = [\boldsymbol{h}_1(0)\ \boldsymbol{h}_1(1)\ \cdots \boldsymbol{h}_1(v_h-1)]^T \tag{A.1}$$

where the k-th column (or row after the transpose operation)

$$\boldsymbol{h}_1(k) = [h_1(kP)\ h_1(kP+1)\ \cdots h_1(kP+P-1)]^T \tag{A.2}$$

contains the P samples corresponding to the k-th symbol interval. Similarly, the CIR for the second channel can be represented as $\boldsymbol{H}_2$. Stacking the P samples of the channel output corresponding to the k-th symbol as a vector

$$\boldsymbol{y}(k) = [y(kP)\ y(kP+1)\ \cdots y(kP+P-1)]^T, \tag{A.3}$$

we can represent the channel output as

$$\boldsymbol{y}(k) = \sum_{i=0}^{v_h-1} \boldsymbol{h}_1(i)s_1(k-i) + \sum_{i=0}^{v_h-1} \boldsymbol{h}_2(i)s_2(k-i) + \boldsymbol{n}_k \tag{A.4}$$

where noise vector $\boldsymbol{n}(k)$ has the same form as (A.3).

To simplify the channel estimation, let us discard the channel observations in a guard interval of length v_h-1 at the beginning of the training sequence. If we place the channel output vectors $\boldsymbol{y}(k), k = v_h, \ldots, L_t$, corresponding to the remaining $L_t - v_h + 1$

symbols, next to each other in a matrix, we can write

$$\boldsymbol{Y} = \boldsymbol{S}\boldsymbol{H} + \boldsymbol{N} = [\boldsymbol{S}_1(v_h, L_t)\ \boldsymbol{S}_2(v_h, L_t)] \begin{bmatrix} \boldsymbol{H}_1 \\ \boldsymbol{H}_2 \end{bmatrix} + \boldsymbol{N} \tag{A.5}$$

where $\boldsymbol{Y}$ and $\boldsymbol{N}$ are of dimension $(L_t - v_h + 1) \times P$, and $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ are Toeplitz matrices of dimension $(L_t - v_h + 1) \times v_h$, and

$$\boldsymbol{S}_i(v_h, L_t) = \begin{bmatrix} s_i(v_h - 1) & \dots & s_i(0) \\ s_i(v_h) & \dots & s_i(1) \\ \vdots & & \vdots \\ s_i(L_t - 1) & \dots & s_i(L_t - v_h) \end{bmatrix} \tag{A.6}$$

for $i = 1, 2$. The block least-squares estimates of the two channels, assuming that $\boldsymbol{S}$ has full column rank, can be calculated as [48]

$$\widehat{\boldsymbol{H}} = \begin{bmatrix} \widehat{\boldsymbol{H}}_1 \\ \widehat{\boldsymbol{H}}_2 \end{bmatrix} = (\boldsymbol{S}^H \boldsymbol{S})^{-1} \boldsymbol{S}^H \boldsymbol{Y}\,. \tag{A.7}$$

If we assume that the noise is additive white complex Gaussian distributed with mean zero and variance $\sigma^2 = E\{|n(k)|^2\}$, the channel estimation MSE is defined to be

$$\text{MSE} = \frac{1}{P}\ E\left\{\text{Tr}\left[(\boldsymbol{H} - \widehat{\boldsymbol{H}})^H (\boldsymbol{H} - \widehat{\boldsymbol{H}})\right]\right\} = \sigma^2\ \text{Tr}\{(\boldsymbol{S}^H \boldsymbol{S})^{-1}\}\,. \tag{A.8}$$

One can show that the theoretical minimum MSE is equal to

$$\text{MMSE} = \frac{\sigma^2 v_h}{L_t - v_h + 1} \tag{A.9}$$

which is achieved if and only if

$$\boldsymbol{S}^H \boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}_1^H \boldsymbol{S}_1 & \boldsymbol{S}_1^H \boldsymbol{S}_2 \\ \boldsymbol{S}_2^H \boldsymbol{S}_1 & \boldsymbol{S}_2^H \boldsymbol{S}_2 \end{bmatrix} = (L_t - v_h + 1)\ \boldsymbol{I}_{2v_h}\,. \tag{A.10}$$

Sequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ satisfying (A.10) are called *optimal* sequences. From (A.10) we see that the optimal sequences have impulse-like autocorrelation and zero cross-

correlations.

As an example, consider letting $L_t = 20$ and $v_h = 5$. If optimal sequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ exist, according to (A.9) they can achieve a MMSE of $0.3125\,\sigma^2$. Optimal sequences can be constructed using an N-th root-of-unity alphabet [49]. Unfortunately, for a QPSK-based system, this involves searching over $4^{20} \times 4^{20}$ different sequences of real and imaginary binary combinations. Since there are over 10^{24} possible combinations of length-20 complex sequence pairs, this exhaustive search is not possible to do in a reasonable amount of time. Due to this prohibitively large search and the advantages of the TR-STBC structure discussed in Sec. 2.2, we instead introduce the suboptimal approach proposed by Fragouli *et al.* in [39].

A.2 TR-STBC-STRUCTURED TRAINING SEQUENCES

Fragouli *et al.* mimic the TR-STBC structure proposed in [1] by dividing each training sequence into two length-$L_t/2$ subsequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$.[1] The first antenna transmits training sequence block $[\boldsymbol{s}_1 \;\; -\widetilde{\boldsymbol{s}}_2^*]$ at the same time that the second antenna transmits $[\boldsymbol{s}_2 \;\; \widetilde{\boldsymbol{s}}_1^*]$. Using these two training sequences consisting of two subsequences, we would like to estimate $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$. A cost of dividing the training into two subsequences is that we need two guard intervals of $v_h - 1$ symbols - one after the first subsequence and one after the second subsequence. If we remove the channel observations corresponding to the first $v_h - 1$ guard symbols for each of the two subsequences and stack the remaining channel observations in the correct manner, we can form the new matrix equation

$$\begin{bmatrix} \boldsymbol{Y}_1 \\ \boldsymbol{Y}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{S}_1 & \boldsymbol{S}_2 \\ -\widetilde{\boldsymbol{S}}_2^* & \widetilde{\boldsymbol{S}}_1^* \end{bmatrix} \begin{bmatrix} \boldsymbol{H}_1 \\ \boldsymbol{H}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{N}_1 \\ \boldsymbol{N}_2 \end{bmatrix} \tag{A.11}$$

[1] We now change the notation slightly. Vectors $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ now represent $L_t/2$-symbol subsequences instead of L_t-symbol sequences.

where $\boldsymbol{Y}_i$ and $\boldsymbol{N}_i$ are of dimension $(L_t/2 - v_h + 1) \times P$, and $\boldsymbol{S}_i$ is of dimension $(L_t/2 - v_h + 1) \times v_h$. Therefore, using the large matrix notation of (A.5), we have

$$\boldsymbol{S}^H \boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}_1^H \boldsymbol{S}_1 + \widetilde{\boldsymbol{S}}_2^T \widetilde{\boldsymbol{S}}_2^* & \boldsymbol{S}_1^H \boldsymbol{S}_2 - \widetilde{\boldsymbol{S}}_2^T \widetilde{\boldsymbol{S}}_1^* \\ \boldsymbol{S}_2^H \boldsymbol{S}_1 - \widetilde{\boldsymbol{S}}_1^T \widetilde{\boldsymbol{S}}_2^* & \boldsymbol{S}_2^H \boldsymbol{S}_2 + \widetilde{\boldsymbol{S}}_1^T \widetilde{\boldsymbol{S}}_1^* \end{bmatrix}. \tag{A.12}$$

As we saw in (A.10), a necessary condition to achieve the lowest possible MSE is to have zero cross-correlation between sequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ or

$$\boldsymbol{S}_1^H \boldsymbol{S}_2 - \widetilde{\boldsymbol{S}}_2^T \widetilde{\boldsymbol{S}}_1^* = \left(\boldsymbol{S}_2^H \boldsymbol{S}_1 - \widetilde{\boldsymbol{S}}_1^T \widetilde{\boldsymbol{S}}_2^* \right)^H = \mathbf{O}\,. \tag{A.13}$$

This condition can be rewritten as

$$\boldsymbol{S}_1^H \boldsymbol{S}_2 = \boldsymbol{J}_{v_h} \left(\boldsymbol{S}_1^H \boldsymbol{S}_2 \right)^T \boldsymbol{J}_{v_h} \tag{A.14}$$

where $\boldsymbol{J}_k = [\delta_{i,k-i+1}]$ denotes the $k \times k$ square reversion ("backward identity") matrix. To satisfy the delta-like autocorrelation requirements on $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, in addition we need

$$\boldsymbol{S}_1^H \boldsymbol{S}_1 - \widetilde{\boldsymbol{S}}_2^T \widetilde{\boldsymbol{S}}_2^* = 2\,(L_t/2 - v_h + 1)\,\boldsymbol{I}_{v_h}\,. \tag{A.15}$$

As shown in [39], two possible solutions to (A.14) and (A.15) are as follows:

1. $\boldsymbol{S}_1^H \boldsymbol{S}_1 = (L_t/2 - v_h + 1)\,\boldsymbol{I}_{v_h}, \widetilde{\boldsymbol{S}}_1 = \boldsymbol{S}_1$, and $\boldsymbol{S}_2 = \boldsymbol{S}_1$. To satisfy these conditions, choose a sequence $\boldsymbol{s}_1$ that is symmetric around its center with impulse-like autocorrelation, and set $\boldsymbol{s}_2 = \boldsymbol{s}_1$.

2. $\boldsymbol{S}_1^H \boldsymbol{S}_1 = (L_t/2 - v_h + 1)\,\boldsymbol{I}_{v_h}$ and $\boldsymbol{S}_2 = \widetilde{\boldsymbol{S}}_1$. To satisfy these conditions, choose a sequence $\boldsymbol{s}_1$ with impulse-like autocorrelation, and set $\boldsymbol{s}_2 = \widetilde{\boldsymbol{s}}_1$.

Since the first solution involves an extra symmetry constraint than the second solution, we choose the second solution for our TR-STBC training sequence. We propose transmitting training sequence blocks $[\boldsymbol{s}_1 \;\; -\boldsymbol{s}_1^*]$ and $[\widetilde{\boldsymbol{s}}_1 \;\; \widetilde{\boldsymbol{s}}_1^*]$.

Returning to our specific example, to find the length-10 training sequence $\boldsymbol{s}_1$, we have written a MATLAB function to search over all 4^{10} complex sequences of the form $\pm 1 \pm i$ to find the sequence which yields the MMSE. We have found this sequence to

be

$$\boldsymbol{s}_1 = [-1-i,\ 1+i,\ -1-i,\ -1+i,\ 1+i,\ 1+i,\ -1+i,\ -1-i,\ -1+i,\ 1-i]\,. \quad \text{(A.16)}$$

Substituting (A.16) into (A.8) we find this TR-STBC structured training sequence achieves a MSE of $0.4851\,\sigma^2$ versus the optimal MMSE $0.3125\,\sigma^2$ achieved by two length-20 N-th root-of-unity sequences. Thus, the cost of the simpler TR-STBC structure is 1.55 times (1.9 dB) the estimation error of the ideal. However, as we show in Sec. 3.2.1, the structure can reduce the hardware complexity, as well as enable an adaptive equalizer to reuse the training data for "learning" the TR-STBC channel.

Appendix B

DERIVATION OF CE-RLS ALGORITHM

In this appendix we derive the Combined Estimate-RLS algorithm from first principles. We would like to find joint filter tap weights $\boldsymbol{w}(n)$ which minimize the weighted least-squares error

$$J_{\mathrm{CE}}(k) = \sum_{i=1}^{k} \lambda^{k-i} \left[|\widetilde{e}_{1i}|^2 + |\widetilde{e}_{2i}|^2 \right] + 2\,\delta\,\lambda^k \|\boldsymbol{w}(k)\|^2 \tag{B.1}$$

where

$$\widetilde{e}_{1i} = s_{1,i-\delta} - \boldsymbol{w}^H(k)\,\boldsymbol{v}_1(i) \qquad \widetilde{e}_{2i} = s_{2,i-\delta} - \boldsymbol{w}^H(k)\,\boldsymbol{v}_2(i)\,. \tag{B.2}$$

As shown in [27], the $(N_f P + N_b) \times (N_f P + N_b)$ time-averaged correlation matrices corresponding to tap input vectors $\boldsymbol{v}_1(i)$ and $\boldsymbol{v}_2(i)$ are, respectively,

$$\begin{aligned} \boldsymbol{\Phi}_1(k) &= \sum_{i=1}^{k} \lambda^{k-i} \boldsymbol{v}_1(i)\,\boldsymbol{v}_1^H(i) + \delta\,\lambda^k \boldsymbol{I} \\ \boldsymbol{\Phi}_2(k) &= \sum_{i=1}^{k} \lambda^{k-i} \boldsymbol{v}_2(i)\,\boldsymbol{v}_2^H(i) + \delta\,\lambda^k \boldsymbol{I}\,. \end{aligned} \tag{B.3}$$

Similarly, the $(N_f P + N_b) \times 1$ time-averaged crosscorrelation vectors are

$$\boldsymbol{z}_1(k) = \sum_{i=1}^{k} \lambda^{k-i} \boldsymbol{v}_1(i)\,s_{1,i-\delta}^* \qquad \boldsymbol{z}_2(k) = \sum_{i=1}^{k} \lambda^{k-i} \boldsymbol{v}_2(i)\,s_{2,i-\delta}^*\,. \tag{B.4}$$

Because, as we have shown in Sec. 3.1.1, the second-order statistics of the outputs of the two filters are identical, we would like to derive the tap weights which solve the

novel "combined estimate" normal equation

$$[\boldsymbol{\Phi}_1(k) + \boldsymbol{\Phi}_2(k)]\, \boldsymbol{w}_{\mathrm{LS}} \stackrel{\triangle}{=} \boldsymbol{\Phi}_{\mathrm{CE}}(k)\, \boldsymbol{w}_{\mathrm{LS}} = [\boldsymbol{z}_1(k) + \boldsymbol{z}_2(k)] \stackrel{\triangle}{=} \boldsymbol{z}_{\mathrm{CE}}(k)\,. \tag{B.5}$$

Following the standard recursive least squares method of solving this equation [27], we first note that the least squares estimates $\boldsymbol{\Phi}_i(k)$ and $\boldsymbol{z}_i(k)$, for $i = 1, 2$, can be computed using the recursive equations

$$\boldsymbol{\Phi}_1(k) = \lambda\, \boldsymbol{\Phi}_1(k-1) + \boldsymbol{v}_1(k)\, \boldsymbol{v}_1^H(k) \tag{B.6}$$

$$\boldsymbol{\Phi}_2(k) = \lambda\, \boldsymbol{\Phi}_2(k-1) + \boldsymbol{v}_2(k)\, \boldsymbol{v}_2^H(k) \tag{B.7}$$

$$\boldsymbol{z}_1(k) = \lambda\, \boldsymbol{z}_1(k-1) + \boldsymbol{v}_1(k)\, s^*_{1,k-\delta} \tag{B.8}$$

$$\boldsymbol{z}_2(k) = \lambda\, \boldsymbol{z}_2(k-1) + \boldsymbol{v}_2(k)\, s^*_{2,k-\delta}\,. \tag{B.9}$$

Adding (B.6) to (B.7) and (B.8) to (B.9), we can write

$$[\boldsymbol{\Phi}_1(k) + \boldsymbol{\Phi}_2(k)] = \lambda\, [\boldsymbol{\Phi}_1(k-1) + \boldsymbol{\Phi}_2(k-1)] + \boldsymbol{v}_1(k)\, \boldsymbol{v}_1^H(k) + \boldsymbol{v}_2(k)\, \boldsymbol{v}_2^H(k) \tag{B.10}$$

$$[\boldsymbol{z}_1(k) + \boldsymbol{z}_2(k)] = \lambda\, [\boldsymbol{z}_1(k-1) + \boldsymbol{z}_2(k-1)] + \boldsymbol{v}_1(k)\, s^*_{1,k-\delta} + \boldsymbol{v}_2(k)\, s^*_{2,k-\delta}\,. \tag{B.11}$$

To solve (B.5), we only need to estimate $\boldsymbol{\Phi}_{\mathrm{CE}}(k)$ and $\boldsymbol{z}_{\mathrm{CE}}(k)$, but not the components $\boldsymbol{\Phi}_i(k)$ and $\boldsymbol{z}_i(k)$, for $i = 1, 2$. Thus, we rewrite the recursions (B.10) and (B.11) as

$$\boldsymbol{\Phi}_{\mathrm{CE}}(k) = \lambda\, \boldsymbol{\Phi}_{\mathrm{CE}}(k-1) + \boldsymbol{V}(k)\, \boldsymbol{V}^H(k) \tag{B.12}$$

$$\boldsymbol{z}_{\mathrm{CE}}(k) = \lambda\, \boldsymbol{z}_{\mathrm{CE}}(k-1) + \boldsymbol{V}(k)\, \boldsymbol{s}^*(k) \tag{B.13}$$

where

$$\boldsymbol{V}(k) = [\boldsymbol{v}_1(k)\ \boldsymbol{v}_2(k)] \qquad \boldsymbol{s}(k) = [s_{1,k-\delta}\ s_{2,k-\delta}]^T\,.$$

To rewrite (B.12), we now invoke the well-known Matrix Inversion Lemma for positive definite symmetric matrices $\boldsymbol{A}$, $\boldsymbol{B}$ with $\boldsymbol{A} = \boldsymbol{B}^{-1} + \boldsymbol{C}\, \boldsymbol{D}^{-1} \boldsymbol{C}^H$. In this case, it can be shown that

$$\boldsymbol{A}^{-1} = \boldsymbol{B} - \boldsymbol{B}\, \boldsymbol{C} (\boldsymbol{D} + \boldsymbol{C}^H \boldsymbol{B}\, \boldsymbol{C})^{-1} \boldsymbol{C}^H \boldsymbol{B}\,.$$

Using the definitions $\boldsymbol{A} = \boldsymbol{\Phi}_{\mathrm{CE}}(k)$, $\boldsymbol{B}^{-1} = \lambda\,\boldsymbol{\Phi}_{\mathrm{CE}}(k-1)$, $\boldsymbol{C} = \boldsymbol{V}(k)$ and $\boldsymbol{D} = \boldsymbol{I}_2$, we find the new recursion

$$\begin{aligned}\boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k) = &\lambda^{-1}\boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k-1) - \lambda^{-2}\boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k-1)\,\boldsymbol{V}(k) \\ &\times \left[\boldsymbol{I}_2 + \lambda^{-1}\boldsymbol{V}^H(k)\,\boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k-1)\,\boldsymbol{V}(k)\right]^{-1}\boldsymbol{V}^H(k)\,\boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k-1)\,. \end{aligned} \tag{B.14}$$

Note that the recursion requires the inversion of the 2×2 matrix in the square brackets. To simplify the notation, setting

$$\begin{aligned}\boldsymbol{P}_{\mathrm{CE}}(k) &= \boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k) \\ \boldsymbol{K}(k) &= \lambda^{-1}\boldsymbol{P}_{\mathrm{CE}}(k-1)\,\boldsymbol{V}(k)\left[\boldsymbol{I}_2 + \lambda^{-1}\boldsymbol{V}^H(k)\,\boldsymbol{P}_{\mathrm{CE}}(k-1)\,\boldsymbol{V}(k)\right]^{-1}, \end{aligned} \tag{B.15}$$

recursion (B.14) becomes

$$\boldsymbol{P}_{\mathrm{CE}}(k) = \lambda^{-1}\boldsymbol{P}_{\mathrm{CE}}(k-1) - \lambda^{-1}\boldsymbol{K}(k)\,\boldsymbol{V}^H(k)\,\boldsymbol{P}_{\mathrm{CE}}(k-1)\,. \tag{B.16}$$

Rearranging (B.15), we see that

$$\begin{aligned}\boldsymbol{K}(k) &= \left[\lambda^{-1}\boldsymbol{P}_{\mathrm{CE}}(k-1) - \lambda^{-1}\boldsymbol{K}(k)\,\boldsymbol{V}^H(k)\,\boldsymbol{P}_{\mathrm{CE}}(k-1)\right]\boldsymbol{V}(k) \\ &= \boldsymbol{P}_{\mathrm{CE}}(k)\,\boldsymbol{V}(k) = \boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k)\,\boldsymbol{V}(k)\,. \end{aligned} \tag{B.17}$$

Returning to the modified normal equation (B.5), the optimal combined estimate filter taps are then

$$\boldsymbol{w}_{\mathrm{CE}}(k) = \boldsymbol{\Phi}_{\mathrm{CE}}^{-1}(k)\,\boldsymbol{z}_{\mathrm{CE}}(k) = \boldsymbol{P}_{\mathrm{CE}}(k)\,\boldsymbol{z}_{\mathrm{CE}}(k)\,. \tag{B.18}$$

Using the update forms of $\boldsymbol{z}_{\mathrm{CE}}(k)$ and $\boldsymbol{P}_{\mathrm{CE}}(k)$ shown in (B.13) and (B.16), we can rewrite $\boldsymbol{w}_{\mathrm{CE}}(k)$ as

$$\begin{aligned}\boldsymbol{w}_{\mathrm{CE}}(k) &= \lambda\,\boldsymbol{P}_{\mathrm{CE}}(k)\,\boldsymbol{z}_{\mathrm{CE}}(k-1) + \boldsymbol{P}_{\mathrm{CE}}(k)\,\boldsymbol{V}(k)\,\boldsymbol{s}^*(k) \\ &= \boldsymbol{P}_{\mathrm{CE}}(k-1)\,\boldsymbol{z}_{\mathrm{CE}}(k-1) - \boldsymbol{K}(k)\,\boldsymbol{V}^H(k)\,\boldsymbol{P}_{\mathrm{CE}}(k-1)\,\boldsymbol{z}_{\mathrm{CE}}(k-1) \\ &\quad + \boldsymbol{P}_{\mathrm{CE}}(k)\,\boldsymbol{V}(k)\,\boldsymbol{s}^*(k)\,. \end{aligned}$$

Applying (B.18) again, we finally have

$$\begin{aligned}\boldsymbol{w}_{\mathrm{CE}}(k) &= \boldsymbol{w}_{\mathrm{CE}}(k-1) - \boldsymbol{K}(k)\,\boldsymbol{V}^H(k)\,\boldsymbol{w}_{\mathrm{CE}}(k-1) + \boldsymbol{P}_{\mathrm{CE}}(k)\,\boldsymbol{V}(k)\,\boldsymbol{s}^*(k) \\ &= \boldsymbol{w}_{\mathrm{CE}}(k-1) + \boldsymbol{K}(k)\left[\boldsymbol{s}^*(k) - \boldsymbol{V}^H(k)\,\boldsymbol{w}_{\mathrm{CE}}(k-1)\right] \\ &= \boldsymbol{w}_{\mathrm{CE}}(k-1) + \boldsymbol{K}(k)\,\boldsymbol{\epsilon}^*(k) \end{aligned} \tag{B.19}$$

where equalizer error vector

$$\boldsymbol{\epsilon}(k) = \boldsymbol{s}(k) - \boldsymbol{V}^T(k)\,\boldsymbol{w}^*_{\mathrm{CE}}(k-1)\,. \tag{B.20}$$

The final form of the CE-RLS algorithm is summarized in Table 3.3.

REFERENCES

[1] E. Lindskog and A. Paulraj, "A transmit diversity scheme for channels with inter-symbol interference," in *Proc. IEEE Int'l. Conf. on Communications*, vol. 1, New Orleans, LA, June 18-22, 2000, pp. 307–311.

[2] N. Al-Dhahir, M. Uysal, and C. N. Georghiades, "Three space-time block-coding schemes for frequency-selective fading channels with application to EDGE," in *Proc. IEEE Vehicular Technology Conf.*, vol. 3, Atlantic City, NJ, Oct. 7-11, 2001, pp. 1834–1838.

[3] N. Al-Dhahir, "Overview and comparison of equalization schemes for space-time-coded signals with application to EDGE," *IEEE Trans. Signal Processing*, vol. 50, pp. 2477–2488, Oct. 2002.

[4] H. Mheidat, M. Uysal, and N. Al-Dhahir, "Comparative analysis of equalization techniques for STBC with application to EDGE," in *Proc. IEEE Vehicular Technology Conf.*, vol. 1, Milan, Italy, May 17-19, 2004, pp. 555–559.

[5] C. Jonietz, W. H. Gerstacker, and R. Schober, "Space-time block coding and receive diversity for WLAN IEEE 802.11b," in *Proc. IEEE Int'l. Conf. on Communications*, vol. 5, Seoul, Korea, May 16-20, 2005, pp. 2890–2895.

[6] J. G. Proakis, *Digital Communications*, 3rd ed. London: McGraw-Hill, 1995.

[7] E. Larsson and P. Stoica, *Space-Time Block Coding for Wireless Communications*. Cambridge, UK: Cambridge University Press, 2003.

[8] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block coding for wireless communications: Performance results," *IEEE J. Select. Areas Commun.*, vol. 17, pp. 451–460, Mar. 1999.

[9] S. N. Diggavi, N. Al-Dhahir, A. Stamoulis, and A. R. Calderbank, "Great expectations: The value of spatial diversity in wireless networks," *Proc. IEEE*, vol. 92, no. 2, pp. 219–270, Feb. 2004.

[10] S. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 1451–1458, Oct. 1998.

[11] J. Dowle, S. H. Kuo, K. Mehrotra, and I. V. McLoughlin, "An FPGA-based MIMO and space-time processing platform," *EURASIP J. Applied Signal Processing*, vol. 2006, pp. 1–14, Article ID 34 653, 2006.

[12] A. Wittneben, "A new bandwidth efficient transmit antenna modulation diversity scheme for linear digital modulation," in *Proc. IEEE Int'l. Conf. on Communications*, Geneva, Switzerland, May 23-26, 1993, pp. 1630–1634.

[13] D. Gesbert, M. Shafi, D.-S. Shiu, P. J. Smith, and A. Naguib, "From theory to practice: An overview of MIMO space-time coded wireless systems," *IEEE J. Select. Areas Commun.*, vol. 21, pp. 281–302, Apr. 2003.

[14] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1456–1467, July 1999.

[15] Z. Liu, G. Giannakis, A. Scaglione, and S. Barbarossa, "Decoding and equalization of unknown multipath channels based on block precoding and transmit-antenna diversity," in *Proc. Asilomar Conf. on Signals, Systems, and Computers*, vol. 2, Oct. 24-27, 1999, pp. 1557–1561.

[16] N. Al-Dhahir, "Single-carrier frequency-domain equalization for space-time block-coded transmissions over frequency-selective fading channels," *IEEE Commun. Lett.*, vol. 7, pp. 304–306, July 2001.

[17] Y. Zhu and K. B. Letaief, "Single-carrier frequency-domain equalization with decision-feedback processing for time-reversal space-time block-coded systems," *IEEE Trans. Commun.*, vol. 53, no. 7, pp. 1127–1131, July 2005.

[18] E. G. Larsson, P. Stoica, and J. Li, "Orthogonal space-time block codes: Maximum likelihood detection for unknown channels and unstructured interferences," *IEEE Trans. Veh. Technol.*, vol. 51, no. 2, pp. 362–372, Feb. 2003.

[19] R. Schober, H. Chen, and W. Gerstacker, "Decision-feedback sequence estimation for time-reversal space-time block-coded transmission," *IEEE Trans. Veh. Technol.*, vol. 53, pp. 1273–1278, July 2004.

[20] K. Mehrotra and I. V. McLoughlin, "Time reversal space time block coding with channel estimation and synchronisation errors," in *Australian Telecommunication Networks and Applications Conference (ANTNAC) 2003*, vol. 1, Melbourne, Australia, Dec. 15-18, 2003, pp. 617–620.

[21] M. Baghaie, S. Kuo, and I. V. McLoughlin, "FPGA implementation of space-time block coding systems," in *IEEE 6th Circ. and Sys. Symposium on Emerging Technologies: Mobile and Wireless Commun.*, vol. 2, Shanghai, China, May 31 - June 2, 2004, pp. 591–594.

[22] I. R. S. Casella, P. J. E. Jeszensky, and E. S. Sousa, "Equalization pos-combining with channel estimation and MIMO joint equalization combining receivers for space-time block coding in frequency selective channels," in *Proc. IEEE Int'l. Symp. on Personal, Indoor, and Mobile Radio Communications*, vol. 2, Beijing, China, Sept. 7-10, 2003, pp. 1317–1321.

[23] Y. Luo and S. Lambotharan, "A new tap constrained constant modulus algorithm for the blind equalisation of time-reversal-based STBCs through the exploitation of algebraic properties," *IEE Proc.-Commun.*, vol. 152, no. 3, pp. 295–301, June 2005.

[24] H. Jafarkhani, "A quasi-orthogonal space-time block code," *IEEE Trans. Commun.*, vol. 49, no. 1, pp. 1–4, Jan. 2001.

[25] S. Haykin, *Communication Systems*, 4th ed. New York: John Wiley and Sons, 2001.

[26] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series, with Engineering Applications.* Cambridge, MA: MIT Press, 1949.

[27] S. Haykin, *Adaptive Filter Theory*, 4th ed. Upper Saddle River, NJ: Prentice-Hall, 2002.

[28] B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE WESCON Convention Record*, vol. 4, pp. 96–104, Aug. 1960.

[29] F. Li, "A convergence rate comparison of QAM and CAP equalizer structures," Masters Thesis, Department of Electrical and Computer Engineering, University of Canterbury, Christchurch, New Zealand, Apr. 2005.

[30] N. J. Bershad, "Analysis of the normalized LMS algorithm with Gaussian inputs," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 34, no. 4, pp. 793–806, Aug. 1986.

[31] J.-I. Nagumo and A. Noda, "A learning method for system identification," *IEEE Trans. Automat. Contr.*, vol. 12, no. 3, pp. 282–287, June 1967.

[32] A. E. Albert and L. S. Gardner, Jr., *Stochastic Approximation and Nonlinear Regression.* Cambridge, MA: MIT Press, 1967.

[33] P. Monsen, "Feedback equalization for fading dispersive channels," *IEEE Trans. Inform. Theory*, vol. 17, pp. 56–64, Jan. 1971.

[34] S. Haykin, *Adaptive Filter Theory*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[35] H. J. Kushner, *Approximation and Weak Convergence Methods for Random Processes with Applications to Stochastic System Theory.* Cambridge, MA: MIT Press, 1984.

[36] G. E. Uhlenbeck and L. S. Ornstein, "The theory of Brownian motion," *Phys. Rev.*, vol. 36, no. 5, pp. 823–841, Sept. 1930.

[37] F. Reif, *Fundamentals of Statistical and Thermal Physics.* New York: McGraw-Hill, 1965.

[38] J. Gao and Y. H. Leung, "A new adaptive equalizer for carrierless amplitude and phase (CAP) receivers," in *Proc. IEEE Int'l. Symp. on Circuits and Systems*, vol. 3, Orlando, FL, May 30-June 2, 1999, pp. 90–93.

[39] C. Fragouli, N. Al-Dhahir, and W. Turin, "Training based channel estimation for multiple-antenna broadband transmissions," *IEEE Trans. Commun.*, vol. 2, pp. 384–391, Mar. 2003.

[40] ITU ITU-R M.1225, "Guidelines for evaluations of radio transmission technologies for IMT-2000," 1997.

[41] T. S. Rappaport, *Wireless Communications: Principles and Practice.* Upper Saddle River NJ: Prentice Hall, 1996.

[42] D. J. Young and N. C. Beaulieu, "The generation of correlated rayleigh random variates by inverse discrete fourier transform," *IEEE Trans. Commun.*, vol. 48, no. 7, pp. 1114–1127, July 2000.

[43] D. Spalding, "CF1 performance specification," Group Research, Tait Electronics Ltd., 558 Wairaki Rd, Christchurch, New Zealand, Tech. Rep. Issue 0.0, Aug. 2003.

[44] S. H. Kuo, "SC TRSTBC architectural specification," Group Research, Tait Electronics Ltd., 558 Wairaki Rd, Christchurch, New Zealand, Tech. Rep. Issue 0.2, July 2003.

[45] M. V. Clark, L. J. Greenstein, W. K. Kennedy, and M. Shafi, "MMSE diversity combining for wide-band digital cellular radio," *IEEE Trans. Commun.*, vol. 40, no. 6, pp. 1128–1135, June 1992.

[46] J.-H. Jang, H.-C. Won, and G.-H. Im, "Cyclic prefixed single carrier transmission with SFBC over mobile wireless channels," *IEEE Signal Processing Lett.*, vol. 13, pp. 261–264, May 2006.

[47] W. M. Younis, A. H. Sayed, and N. Al-Dhahir, "Efficient adaptive receivers for joint equalization and interference cancellation in multiuser space-time block-coded systems," *IEEE Trans. Signal Processing*, vol. 11, pp. 2849–2862, Nov. 2003.

[48] S. N. Crozier, D. D. Falconer, and S. A. Mahmoud, "Least sum of squared errors LSSE channel estimation," *IEE Proc. F: Radar, Sonar Navig.*, vol. 138, no. 4, pp. 371–378, Aug. 1991.

[49] D. C. Chu, "Polyphase codes with good correlation properties," *IEEE Trans. Inform. Theory*, vol. 18, no. 4, pp. 531–532, July 1972.